Kozio® Host Tools

User Manual

Product names mentioned in this document are trademarks of their respective manufacturers and are used here only for identification purposes.

© Copyright 2011, Kozio, Inc. All Rights Reserved.

Information in this document is subject to change without notice. The software described in this document is furnished under a license agreement. The software may be used, copied or distributed only in accordance with that agreement. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or any means, electronically or mechanically, including photocopying and recording for any purpose without the written permission of Kozio Inc.

HS-DOC-UM-41-R1-2011-DJ

Contents

1 Introduction to Kozio® Host Tools Software 5
1.1 Quick Start Instructions 6
2 Installation 7
2.1 DUT Installation Instructions 7
2.2 Host Tool Installation Instructions for Windows 7
2.3 Windows Firewall Instructions 13
2.4 Updating a Previous Version of the Host Tools 15
3 Communications Setup – Host Computer to DUT 19
3.1 kDiagnostics® Remote Communications 19
3.1.1 Displaying Remote Communications Variables 20
3.1.2 Enabling kDiagnostics® Remote Communications 20
3.1.3 Establishing Ethernet Communications to the DUT 21
3.2 Establishing an Optional SLIP Communications to the DUT 21
3.2.1 Creating a SLIP Connection under Windows XP 21
3.2.2 SLIP — Creating a New Network Connection 21
3.2.3 SLIP — Create a New Direct Connection 22
3.2.4 SLIP — Setting the Kozio Target Properties 22
3.2.5 SLIP — Establishing a "Kozio Target" Network Connection 22
3.2.6 SLIP — Restarting the Windows SLIP Connection 23
3.2.7 SLIP — Verifying the Windows SLIP Connection 23
3.3 Kozio Properties Files 24
3.3.1 KozioServer Properties File Variables 26
3.3.2 SequenceRunner Properties File Variables 26
3.3.3 Restarting KozioServer™ 29
4 ValidationAssistant™ 30
4.1 Getting Started 30
4.1.1 Start-Up Banner 30
4.1.2 Welcome Screen 31
4.1.3 Screen Overview 32
4.2 DUT Status 32
4.3 Validating Setup 33
4.4 Menu Bar 33
4.4.1 File Menu 33
4.4.1.1 Log DUT Output 33
4.4.1.2 Log Commands 34
4.4.1.3 Load Command Tree 34
4.4.1.4 Exit 34
4.4.2 Edit Menu 34
4.4.2.1 Clear Window 34
4.4.3 DUT Menu 34
4.4.3.1 Stop DUT Command 34
4.4.3.2 Download File to DUT 35
4.4.3.3 Execute Script 36
4.4.4 Help Menu 37
4.5 Status Line Indicators 37
4.6 Commands 37
4.6.1 Executing Commands 38
4.7 Command Tree 38

4.7.1 Load Command Tree 38
4.7.2 Executing Tests through the Command Tree 39
4.7.3 Command Tree Prompt 39
4.8 Command Line 39
4.9 Low-Level Commands 40
4.9.1 Low-Level Help 40
4.10 Test Methods 41
4.11 Performance Tests 41
4.12 Diagnostic Tests 41
4.13 Test Suites 42
4.13.1 Test Suites Tree 43
4.13.2 Test Suites Report 43
4.14 Troubleshooting Test Failures 43
4.14.1 Troubleshooting with Test Methods 43
4.14.2 Troubleshooting with Tracing 43
4.14.3 Troubleshooting with Looping 45
4.14.4 Memory Test 46
4.15 Downloading Scripts 46
4.16 TFTP File Transfer 47
4.17 Shortcuts, Keyboard 47
5 SequenceRunner™ 48
5.1 Launching SequenceRunner™ 48
5.2 Log In 49
5.3 Executing Test Sequences 49
5.4 Detailed Log 52
5.5 Command Line Interface (CLI) 53
5.6 CLI Return Value 53
5.7 CLI Options 54
5.8 Executing Commands with CLI 54
5.9 Executing a Test Script with CLI 54
5.10 Executing a Test Sequence with CLI 55
Report Files 56
5.11 SQL Database 57
5.12 XML Validation 57
5.13 Test Sequence Files 57
5.13.1 TestSequence Tag 58
5.13.2 Action Tags 58
5.13.2.1 Connect 59
5.13.2.2 Disconnect 59
5.13.2.3 DownloadImage 59
5.13.2.4 DownloadScript 60
5.13.2.5 ExecuteCommand 60
5.13.2.6 GetDatabase 62
5.13.2.7 GetFile 62
5.13.2.8 GetValue 63
5.13.2.9 Message 64
5.13.2.10 OperatorComment 64
5.13.2.11 Prompt 66
5.13.2.12 PutDatabase 67
5.13.2.13 PutFile 68

5.13.2.14 ScanBarcode....68
5.13.2.15 ScanLabel....69
5.13.2.16 SetSerialNumber....70
5.13.2.17 SystemCommand....70
5.13.3 Passing Values between Test Sequence Actions....71
6 SequenceReporter™....72
6.1 Switching Database Settings....73
6.2 Generate Report....74
6.3 Print Query Results....75
6.4 Save Report....76
7 Multi-device Testing....79
7.1 Multi-device Testing Equipment Virtual Fixture Set Up Example....79
7.2 Software Installation....79
7.3 Properties File....79
7.4 SequenceRunner™....80
8 kHeader....81
9 Troubleshooting....83
9.1 Timeout on command acknowledgement from DUT....83
9.2 Windows Security Alert....84
9.3 Windows Firewall Issues....84
9.4 License Errors....84
10 Best Practices, Acronyms & Glossary....86
10.1 Best Practices....86
10.2 Acronyms....86
10.3 Glossary....86
11 Contact Information....88

1
Introduction to Kozio® Host Tools Software

Kozio's Host Tools consists of programs that provide a simple yet powerful user interface to Kozio's kDiagnostics®.

For engineering, the Host Tools streamline the creation, execution, and automated recording of board-level test sequences used to quickly validate, troubleshoot and program hardware devices. The Host Tools offer an effective means of providing in-system programming.

For manufacturing, the Host Tools efficiently validate and serialize devices during the manufacturing and support phases of a product's lifecycle. Manufacturers save time and money while increasing test coverage and gaining efficiency in design and manufacturing processes.

Kozio's integrated In-System Diagnostics solution includes:

kDiagnostics® - The foundational component of Kozio's complete In-system Diagnostics platform. It is a binary application that executes from the CPU on your embedded system. It contains a test and diagnostics interpreter and executive that runs on your target hardware and executes tests under user control from a test library.

ValidationAssistant™ - a remote interactive hardware debugging and fault isolation tool for engineers working with embedded hardware systems. ValidationAssistant™ runs on any computer in the network and interacts with a test executive running directly on the Device Under Test.

kDev™ - Allows a developer to extend the capabilities of kDiagnostics by compiling and linking in custom source code test routinesinto new kDiagnostics binaries.

kPOST™ - Permits kDiagnostics users to ship Kozio's high-integrity, high-coverage test capabilities with their finished products, enabling a full hardware check to be performed at every power cycle.

SequenceRunner™ –Test operator's interface for automated testing and configuration of embedded systems including operator prompting, real-time functional tests, device programming, file transfers, unique id automation and integrated SQL database for test tracking.

SequenceRunnerCLI™ - provides the power of ValidationAssistant and SequenceRunner in a simple command line-oriented program, perfect for power users, scripting, and integrating Kozio software with your existing production test process.

SequenceReporter™ –Test reporting and tracking used to report the status of board testing by a variety of criteria.

KozioServer™ - an application that connects all clients to multiple targets simultaneously. KozioServer maintains the direct connection to the target.

Requirements for using the Kozio Host Tools:

- Windows XP, Windows Vista, or Windows 7.
- If you are using Windows Vista or Windows 7, Microsoft removed support for SLIP (Serial Line IP). As a result, if you are using SLIP and a serial connection from the Kozio host tools to a DUT

you must find a SLIP connection or install the Windows XP emulation mode which still offers SLIP.

- Release 2008.2 or later of kDiagnostics®

NOTE: Execution of any of ValidationAssistant™ requires the acquisition of a license.

Once purchased, a license can be obtained by going to www.kozio.com/license and providing information that validates the purchase of Kozio software.

1.1 Quick Start Instructions

The steps below provide a brief description of the steps that must be taken to get Kozio's automated design and manufacturing tests running on the test station.

	On the Device Under Test (DUT)	Section	On the Host	Section
1)	Program or load kDiagnostics® on to the DUT. (If not already running on the DUT).	2.1		
2)	Configure the kDiagnostics® remote communications variables using kDiagnostics and a terminal emulator	3.1		
3)	Enable kDiagnostics® remote communication using the kDiagnostics command "kremote.run". (This may be done automatically by creating a kDiagnostics binary that runs this command automatically.)	3.1.2		
4)			Install the host tools using kozio-install-va.exe to install all products or use kozio-install-va-eval.exe to only install ValidationAssistant	2.2
5)			Add exceptions to Windows Firewall settings	2.3
6)			**[Optional]** Establish a SLIP connection to the DUT if using a serial connection.	3.2.1
7)			Run ValidationAssistant, SequenceRunner or SequenceReporter	4.1
8)			UsingValidationAssistant™ type in "version" in the command line to verify quickly that all steps have been completed properly.	4.3

2
Installation

The Host Tools installation program installs the tool on one or more host computers for use by engineering, manufacturing and support. The execution requires the installation of kDiagnostics® on the Device Under Test (DUT). ValidationAssistant interacts with kDiagnostics® through KozioServer™.

KozioServer™ can run from the same computer as ValidationAssistant or from any computer that is network accessible. When using a direct connection, such as serial, KozioServer must be installed on the computer connected to the DUT.

2.1 DUT Installation Instructions

There are two main options for executing kDiagnostics; either program the self-booting kDiagnostics® image into the boot memory on the DUT; or use an external loader to load the kDiagnostics® binary image that executes directly from RAM. A boot loader or JTAG interface can be used to load and run kDiagnostics® from RAM on the device under test.

This manual does not provide instructions for installing kDiagnostics® on the DUT. However; kDiagnostics® setup instructions are included in section 3.1.2 of the **kDiagnostics® User Manual** which is a separate document provided through your private FTP site.

2.2 Host Tool Installation Instructions for Windows

Execute the Host Tools installer executable provided. There are several install files available. The installer for ValidationAssistant only is:

```
kozio-install-va-RxxByy.exe
```

The installer for the full product suite is:

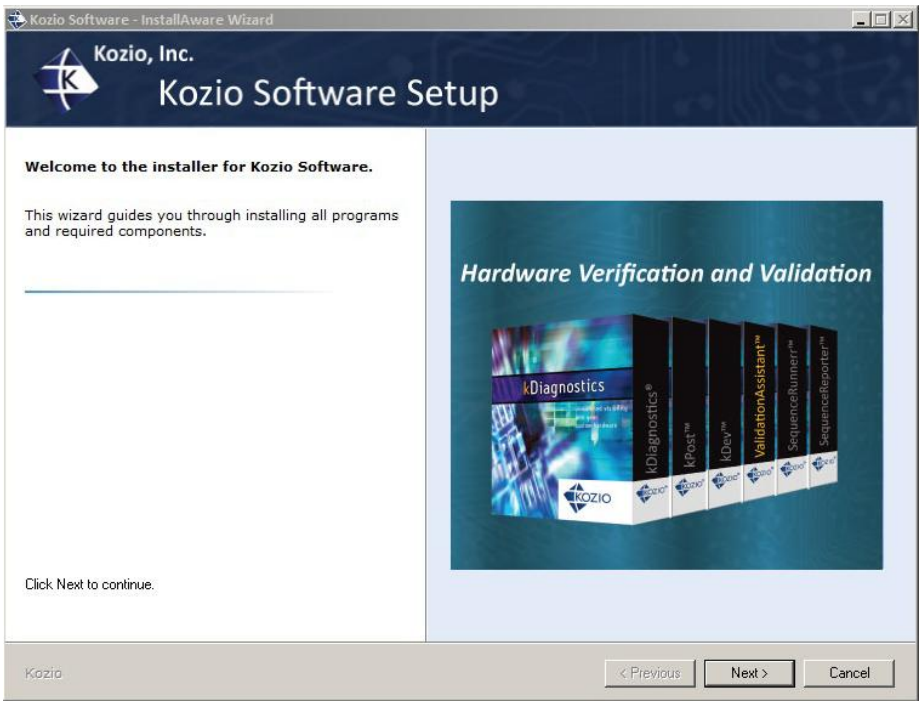
kozio-install-sw-RxxByy.exe

Note: RxxByy indicates the version, for example R41B19 is Release 4.1 Build 19.

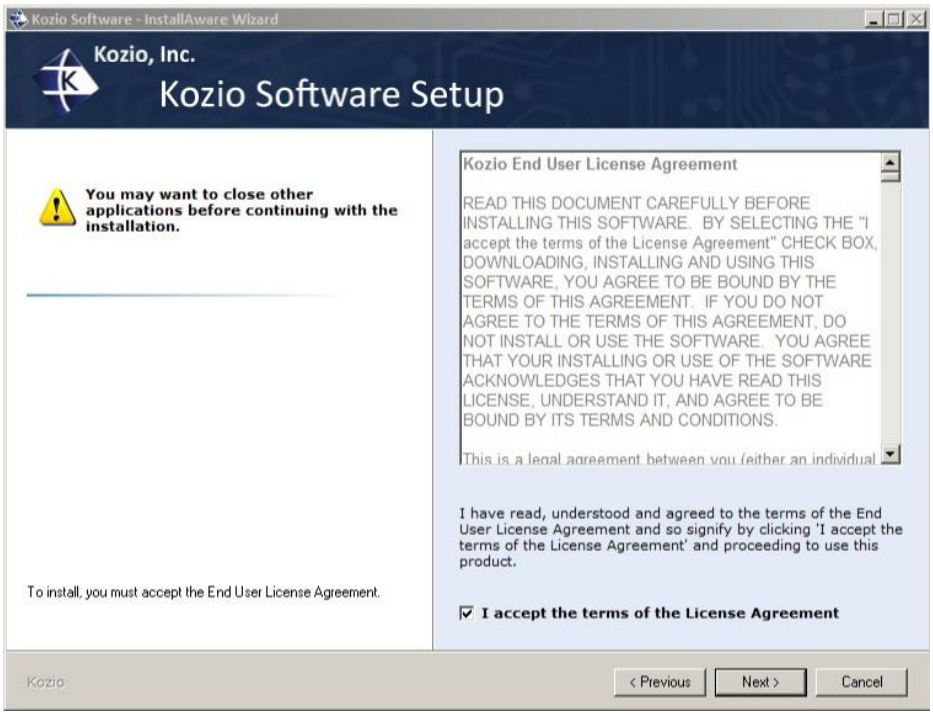
NOTES:

- If installing only ValidationAssistant on a single computer, you can use "`kozio-install-va-eval-RxxByy.exe`". This installer installs KozioServer and ValidationAssistant and uses default configuration parameters thereby simplifying the installation process.
- The JAVA JRE is required to run the Host Tools. The installer will attempt to install the latest version if it does not exist on the host computer.
- Adding Kozio exceptions to Windows Firewall is critical for proper execution of ValidationAssistant. Instructions are provided in Section 2.3.
- Be sure to save the installer file for future use. You will need it if you wish to upgrade the software to a newer version or if you wish to remove the software.

1. Launch the install file.



2. Select "***Next***".
3. Read the Kozio End User License Agreement.

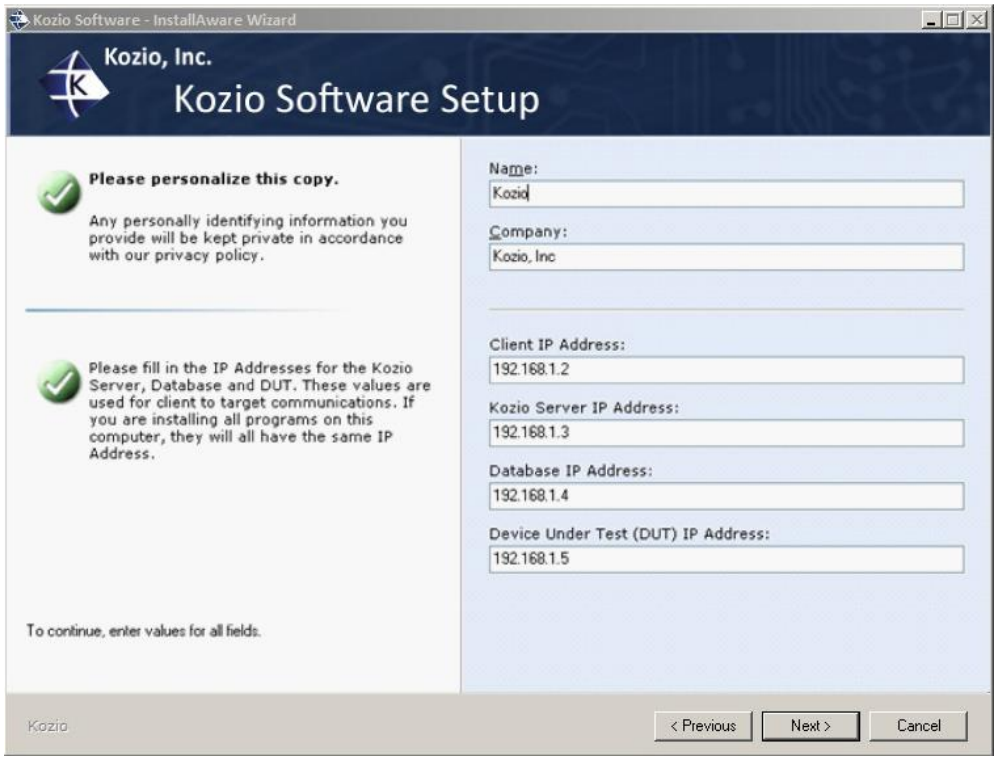

Kozio Software - InstallAware Wizard
Kozio, Inc.
Kozio Software Setup
You may want to close other applications before continuing with the installation.
To install, you must accept the End User License Agreement.
Kozio End User License Agreement
READ THIS DOCUMENT CAREFULLY BEFORE INSTALLING THIS SOFTWARE. BY SELECTING THE "I accept the terms of the License Agreement" CHECK BOX, DOWNLOADING, INSTALLING AND USING THIS SOFTWARE, YOU AGREE TO BE BOUND BY THE TERMS OF THIS AGREEMENT. IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, DO NOT INSTALL OR USE THE SOFTWARE. YOU AGREE THAT YOUR INSTALLING OR USE OF THE SOFTWARE ACKNOWLEDGES THAT YOU HAVE READ THIS LICENSE, UNDERSTAND IT, AND AGREE TO BE BOUND BY ITS TERMS AND CONDITIONS.
This is a legal agreement between you (either an individual
I have read, understood and agreed to the terms of the End User License Agreement and so signify by clicking 'I accept the terms of the License Agreement' and proceeding to use this product.
I accept the terms of the License Agreement
Kozio
< Previous
Next >
Cancel

4. Select the "***I accept the terms of the License Agreement***" check box.

5. Select "***Next***" to continue.

6. In the setup dialog box below, enter your name, company name and very important IP addresses.
   a. The "**Client IP Address**" must contain the IP address of the computer where the Host Tools are installed.
   b. The "**KozioServer IP Address**" must contain the IP address of the computer where KozioServer™ is installed. Use the command "`ipconfig /all`" in a Microsoft Windows Command Prompt to list a computer's IP address.
   c. The "**DUT IP Address**" may change but put in a valid IP address to start. See Section 3.3 for information on how to update these values after installation.

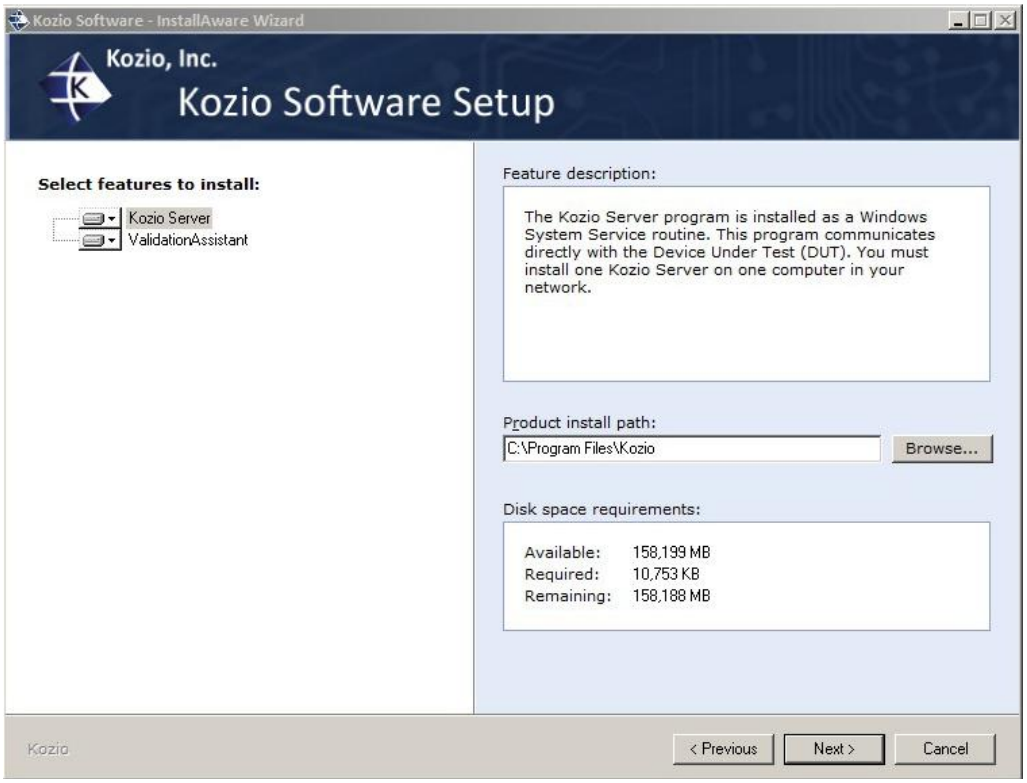
NOTE: The entries shown below are examples only. Provide values that are valid for your network configuration. This dialog only appears when using the full product installer.


Kozio Software - InstallAware Wizard
Kozio, Inc.
Kozio Software Setup
Please personalize this copy.
Any personally identifying information you provide will be kept private in accordance with our privacy policy.
Please fill in the IP Addresses for the Kozio Server, Database and DUT. These values are used for client to target communications. If you are installing all programs on this computer, they will all have the same IP Address.
To continue, enter values for all fields.
Name:
Kozio
Company:
Kozio, Inc
Client IP Address:
192.168.1.2
Kozio Server IP Address:
192.168.1.3
Database IP Address:
192.168.1.4
Device Under Test (DUT) IP Address:
192.168.1.5
Kozio
< Previous
Next >
Cancel

7. Select "***Next***".

8. The dialog box below selects which programs will be installed on the host computer. Select a feature on the left panel and the feature description is updated.

   Use the right-mouse menu over a feature to exclude or include a feature. The installation path may also be changed, but it is recommended to select the default path, C:\Program Files\Kozio.


Kozio Software - InstallAware Wizard
Kozio, Inc.
Kozio Software Setup
Select features to install:
Kozio Server
ValidationAssistant
Feature description:
The Kozio Server program is installed as a Windows System Service routine. This program communicates directly with the Device Under Test (DUT). You must install one Kozio Server on one computer in your network.
Product install path:
C:\Program Files\Kozio
Browse...
Disk space requirements:
Available: 158,199 MB
Required: 10,753 KB
Remaining: 158,188 MB
Kozio
< Previous
Next >
Cancel

9. Select "***Next***".

10. You can enter the name of the Windows Start Menu group identifier in the dialog box below but we suggest using the default value.

Kozio Software - InstallAware Wizard
Kozio, Inc.
Kozio Software Setup
Choose the Start Menu group.
A new program group will be created in your Start Menu to contain product icons. You may create icons for all users to see, or just for yourself.
Start Menu group:
Kozio Suite
Create shortcuts for:
All users of this computer
All users of this computer can see and use the product shortcuts
Just me
Only you may see the product shortcuts
Kozio
< Previous
Next >
Cancel

11. Select "***Next***".

    The system is now ready to be updated. The "***Previous***" button returns to the previous page to allow modification any of the parameters specified prior to completing the installation.

12. Select "***Next***".

13. The Installer will now install all Host Tools selected.

14. When the Kozio Host Tools installation is finished, the final Kozio installation dialog box is displayed.

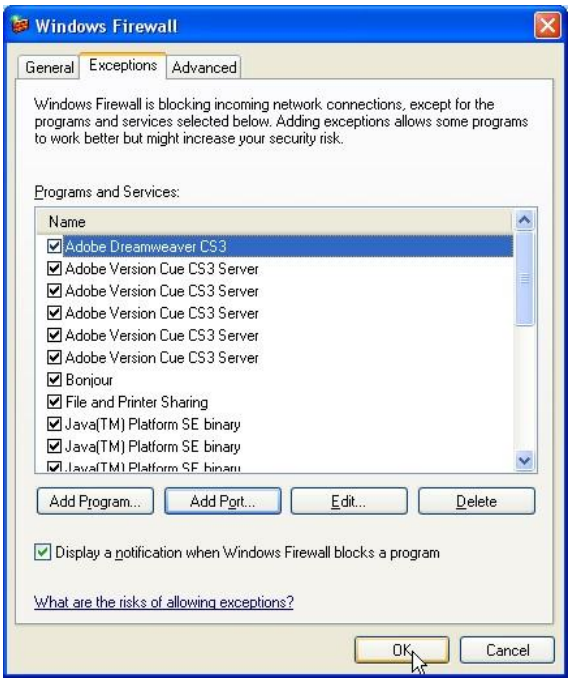
15. When the Kozio license has been received, be sure to copy the license file into the installation directory (recommended location: C:\Program Files\Kozio).

2.3 Windows Firewall Instructions

The latest Service Packs from Microsoft automatically enable the Windows Firewall feature. This feature adds a level of security to the host computer. One feature of the firewall is that many TCP and UDP ports are blocked unless specific access is allowed. ValidationAssistant uses TCP and UDP ports for communications.

If Windows Firewall is used, and it may be automatically enabled, the host computer must be configured for access in order to receive communications over four specific ports. Without this access, the Kozio software cannot function properly. This step is only required once, even if Kozio software is uninstalled and reinstalled.

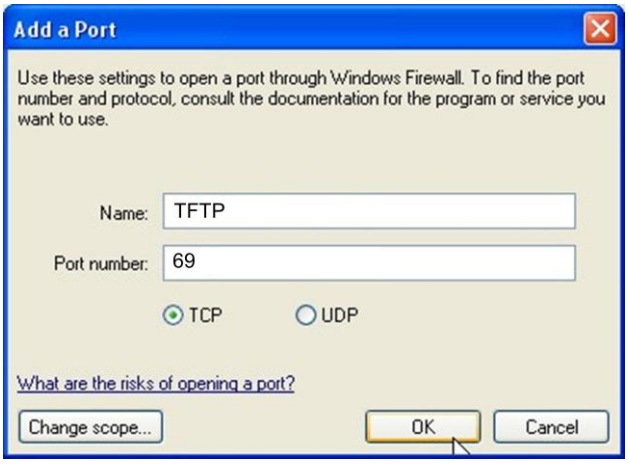
Follow these steps to enable the necessary ports.


Windows Firewall
General
Exceptions
Advanced
Windows Firewall is blocking incoming network connections, except for the programs and services selected below. Adding exceptions allows some programs to work better but might increase your security risk.
Programs and Services:
Name
Adobe Dreamweaver CS3
Adobe Version Cue CS3 Server
Adobe Version Cue CS3 Server
Adobe Version Cue CS3 Server
Adobe Version Cue CS3 Server
Adobe Version Cue CS3 Server
Bonjour
File and Printer Sharing
Java(TM) Platform SE binary
Java(TM) Platform SE binary
Add Program...
Add Port...
Edit...
Delete
Display a notification when Windows Firewall blocks a program
What are the risks of allowing exceptions?
OK
Cancel

1. From the Windows task bar, select "***Start***"
2. Select "***Control Panel***"
3. Double-click "***Windows Firewall***"
4. Select the "***Exceptions***" tab
5. Select the "***Add Port…***" button
6. Execute these four steps to add all four ports.

   a. Select the "***Add Port…***" button. Type "**TFTP**" into the "***Name:***" text box. Type "**69**" into the "***Port number:***" text box. Select the "***UDP***" radio button. Select OK.

      This step enables Ethernet communications for TFTP file transfers and image downloads.


Add a Port
Use these settings to open a port through Windows Firewall. To find the port number and protocol, consult the documentation for the program or service you want to use.
Name:
TFTP
Port number:
69
TCP
UDP
What are the risks of opening a port?
Change scope...
OK
Cancel

   b. Select the "***Add Port…***" button. Type "**Kozio Server Port 1**" into the "***Name:***" text box. Type "**4333**" into the "***Port number:***" text box. Select the "***UDP***" radio button. Select OK.

      This step enables Ethernet communications between the DUT and Kozio Server. This port number must match the *kozioserver.properties* file settings and 4333 is the default value.

   c. Select the "***Add Port…***" button. Type "**Kozio Server Port 2**" into the "***Name:***" text box. Type "**4334**" into the "***Port number:***" text box. Select the "***UDP***" radio button. Select OK.

This step enables Ethernet communications between the DUT and KozioServer™. This port number must match the *kozioserver.properties* file settings. The port number is 4333+1, the initial port plus one.

d. Select the "***Add Port…***" button. Type "**Kozio RMI**" into the "***Name:***" text box. Type "**1099**" into the "***Port number:***" text box. Select the "***UDP***" radio button. Select OK.

This step enables Ethernet communications between the clients and KozioServer™.

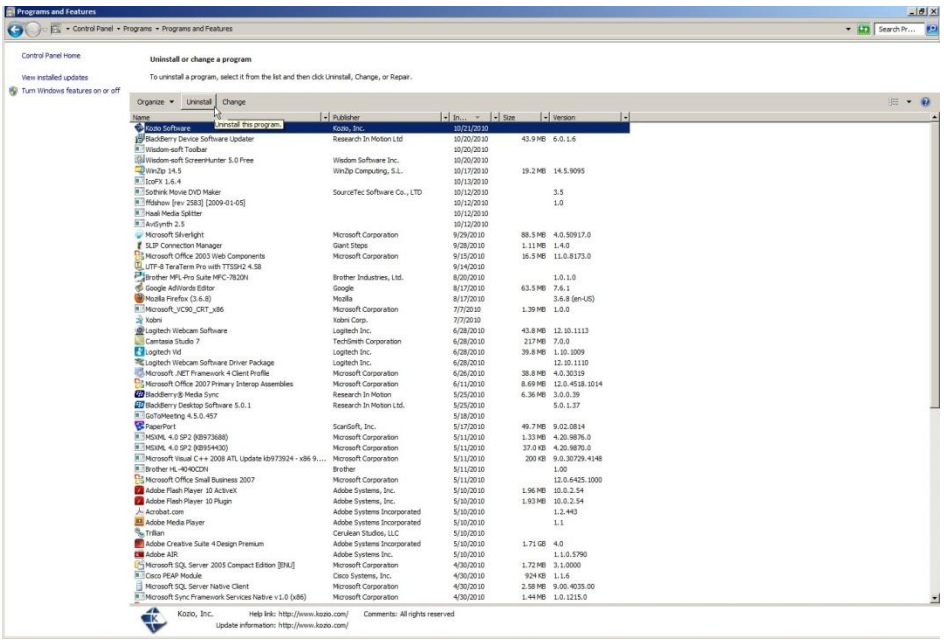
7. Select "***OK***" to close the "***Windows Firewall***" dialog box.

2.4 Updating a Previous Version of the Host Tools

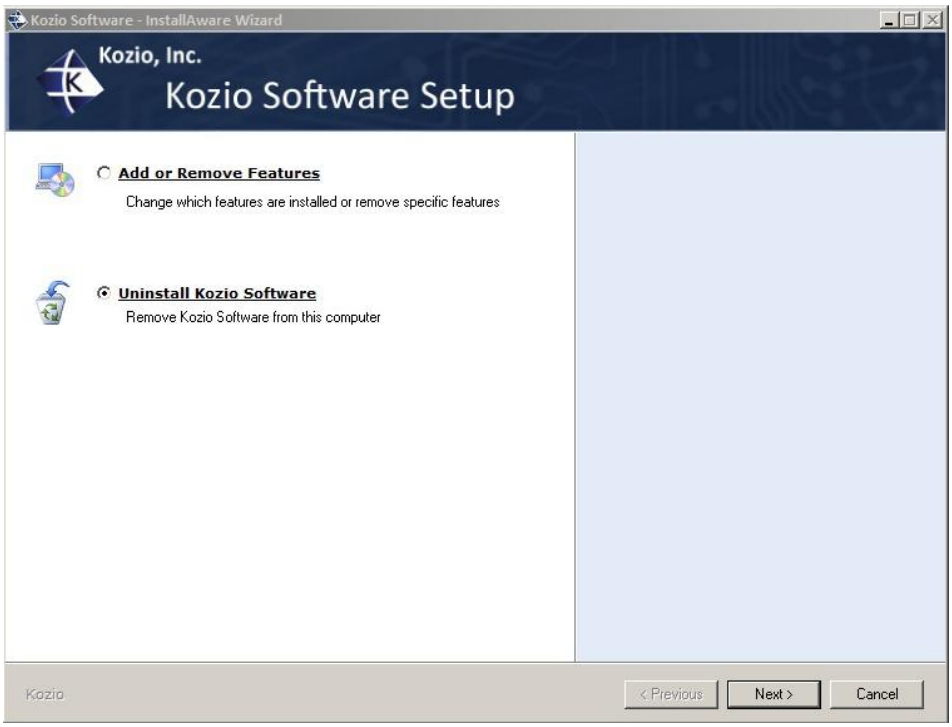
To install or update a new version of the Host Tools, back up all test sequence and log files, and then uninstall the current version. **NOTE**: you will need the install file that you used to install the previous version of the software. The uninstall procedure uses this file and will not work without it.

To uninstall, follow these steps:

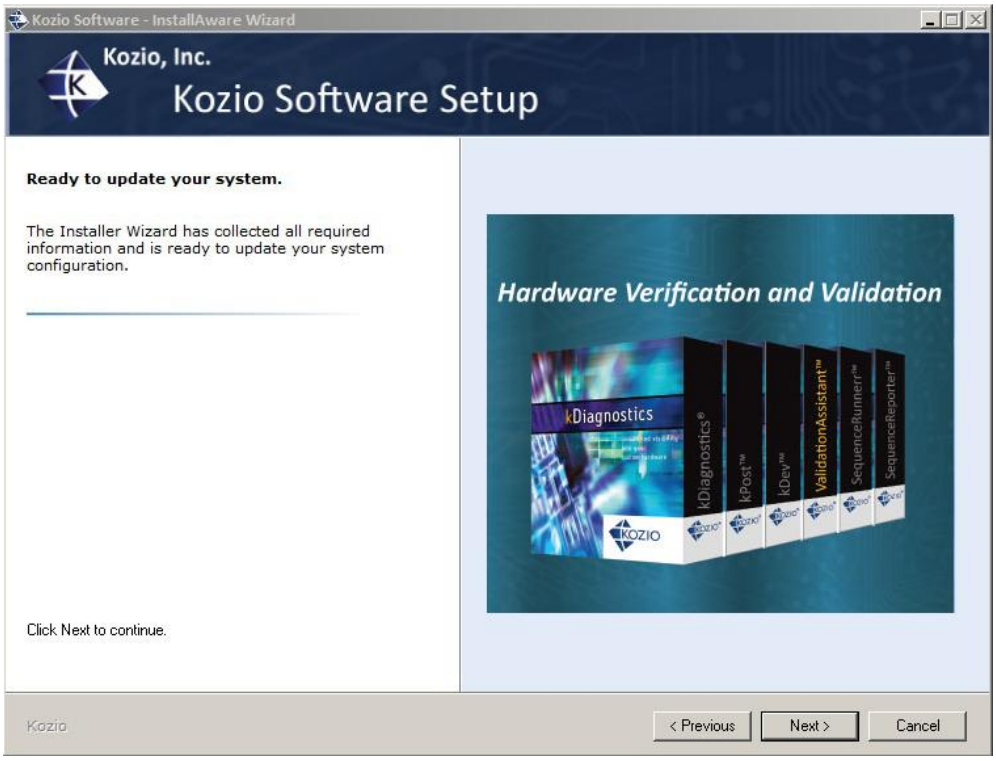
1. From the Windows task bar, select "***Start***"
2. Select "***Control Panel***"
3. Double-click "***Add or Remove Programs***"
4. Scroll down and select the "***Kozio Suite***" item



5. Select the "***Uninstall***" radio button



6. The Kozio Installer will launch. Select the "***Uninstall Kozio Suite***" radio button and then select "***Next***".



7. Select "***Next***" to begin uninstalling the Kozio Host Tools.

8. After the process completes, select "***Finish***" closing the last dialog box.

NOTE: This process will not uninstall the Java Runtime Environment (JRE). If uninstalling those items is necessary, return to Windows "***Start / Control Panel / Add or Remove Program***" and remove these applications from that dialog box.

It is important to note that the install file should not be deleted as it will be needed for upgrading or removing the product.

Install the new version of the Kozio Host Tools by running the latest Kozio installer as described in Section 2.2 Host Tool Installation Instructions. Review the test and configuration files that were backed up. Merge or overwrite changes with the new files as required.

3
Communications Setup – Host Computer to DUT

This section provides information on how to configure the DUT and adjust KozioServer™ properties as necessary to allow communication between the Host Tools and the DUT.

With the Kozio Host Tools installed, the host computer is now ready to communicate with the DUT. Before beginning a testing sequence, a kDiagnostics® command to enable DUT and Host Tools communication MUST be executed.

The required kDiagnostics® command enables a communication protocol between kDiagnostics® and KozioServer™. This command may be executed automatically through a kScript file built into the DUT image or manually from the kDiagnostics® command line.

kDiagnostics® is typically delivered with the remote communications protocol (kRemote) automatically enabled. To manually manage the remote communications protocol, modify the communications settings as described below and then start the kRemote communications protocol on the DUT.

NOTE: KozioServer™ is automatically launched during the Host Tools installation. Details about changing server configuration properties and restarting KozioServer™ to work with the network configuration changes are provided in section 3.3. The Host Tools, SequenceRunner™, ValidationAssistant™, or the SequenceRunner/CLI can be used to remotely communicate with the DUT.

3.1 kDiagnostics® Remote Communications

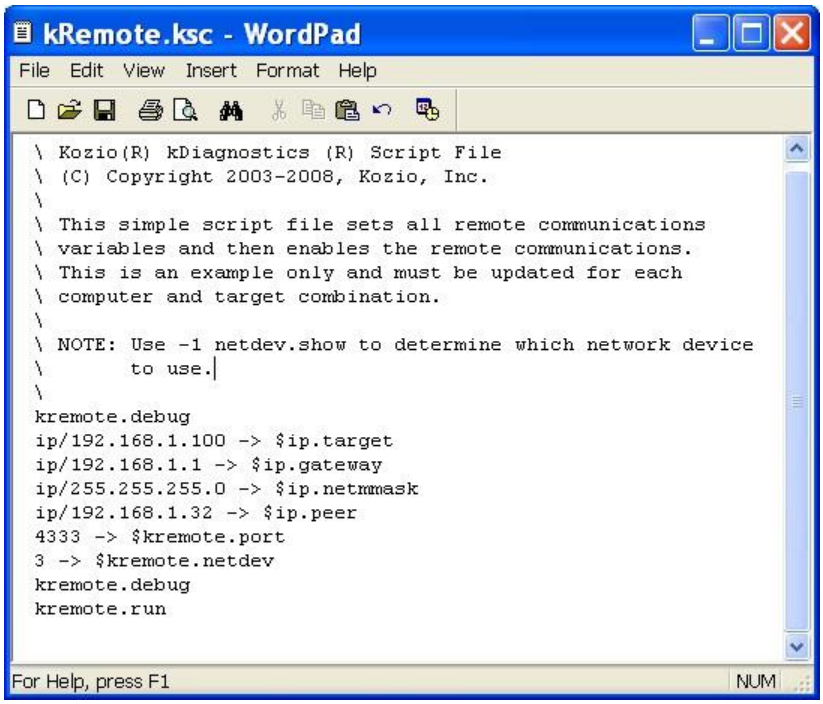
The variables used by kRemote are listed in the following table. Before starting kRemote you should review each variable setting and make any changes that may be necessary as described in 3.1.1.

Variable	Example	Description	To Update
$ip.target	192.168.1.210	The IP address of the DUT. This value is required by KozioServer™.	ip/192.168.1.210 → $ip.target
$ip.gateway	192.168.1.0	The network gateway used by the DUT.	ip/192.168.1.0 → $ip.gateway
$ip.netmask	255.255.255.0	The network mask used by the DUT.	ip/255.255.255.0 → $ip.netmask
$kremote.port	4333	The TCP/UDP port number used for communications between the DUT and server.	4333 -> $kremote.port
$kremote.netdev	3	The network communications device that the DUT will use for remote communications. **Note**: Use "-1 netdev.show" to list all available network devices.	3 → $kremote.netdev

The last column describes how to update a given variable. For IP address updates, the "ip/" precedes the new IP address in dot notation. The right arrow (→), a minus sign followed immediately by a greater than sign, denotes a variable assignment. The last item is the variable name to be updated.

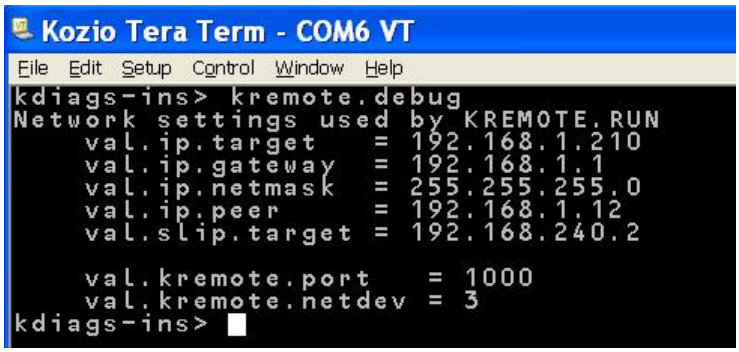
Update any variable by using the kDiagnostics® command line or save the commands in a text file and have kDiagnostics parse the file.

Kozio uses the "*.ksc" extension to denote a kScript test file. Files of this type are ASCII text files containing Kozio kScript commands. Once saved in a file, commands can be built into an updated kDiagnostics® binary image file or parsed in real time by kDiagnostics. The script file can be sent to kDiagnostics® using ValidationAssistant, terminal emulation software (such as TeraTerm), or to an associate via email.


kRemote.ksc - WordPad
File Edit View Insert Format Help
\ Kozio(R) kDiagnostics (R) Script File
\ (C) Copyright 2003-2008, Kozio, Inc.
\
\ This simple script file sets all remote communications
\ variables and then enables the remote communications.
\ This is an example only and must be updated for each
\ computer and target combination.
\
\ NOTE: Use -1 netdev.show to determine which network device
\ to use.
\
kremote.debug
ip/192.168.1.100 -> $ip.target
ip/192.168.1.1 -> $ip.gateway
ip/255.255.255.0 -> $ip.netmmask
ip/192.168.1.32 -> $ip.peer
4333 -> $kremote.port
3 -> $kremote.netdev
kremote.debug
kremote.run
For Help, press F1
NUM

3.1.1 Displaying Remote Communications Variables

Use "**kremote.debug**" to display current remote communication variables and their values.



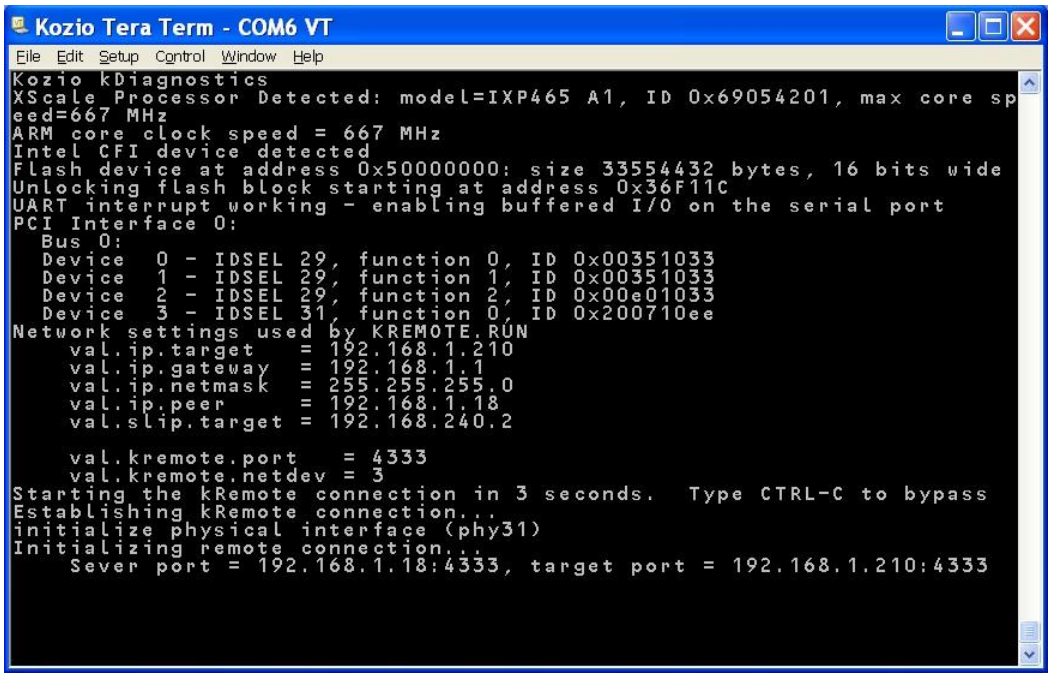
3.1.2 Enabling kDiagnostics® Remote Communications

Use the command "`kremote.run`" to enable the communications between kDiagnostics® and KozioServer™. This kDiagnostics® command relies on a number of environment variables for configuring network parameters as described in Section 3.1.

NOTE: This step is not required if kDiagnostics® has been built to automatically run the "kremote.run" command.

After "`kremote.run`" is executed, the remote communications is established, the terminal emulation window can no longer be used for communication with the DUT. Verify the connection by issuing a host ICMP ping command to the DUT IP address, which is stored in the kDiagnostics® variable "$ip.target".

The following screen capture depicts what would be displayed by kDiagnostics® on a terminal emulation screen after the "`kremote.run`" command is executed.


Kozio Tera Term - COM6 VT
File Edit Setup Control Window Help
Kozio kDiagnostics
XScale Processor Detected: model=IXP465 A1, ID 0x69054201, max core speed=667 MHz
ARM core clock speed = 667 MHz
Intel CFI device detected
Flash device at address 0x50000000: size 33554432 bytes, 16 bits wide
Unlocking flash block starting at address 0x36F11C
UART interrupt working - enabling buffered I/O on the serial port
PCI Interface 0:
Bus 0:
Device 0 - IDSEL 29, function 0, ID 0x00351033
Device 1 - IDSEL 29, function 1, ID 0x00351033
Device 2 - IDSEL 29, function 2, ID 0x00e01033
Device 3 - IDSEL 31, function 0, ID 0x200710ee
Network settings used by KREMOTE.RUN
val.ip.target = 192.168.1.210
val.ip.gateway = 192.168.1.1
val.ip.netmask = 255.255.255.0
val.ip.peer = 192.168.1.18
val.slip.target = 192.168.240.2
val.kremote.port = 4333
val.kremote.netdev = 3
Starting the kRemote connection in 3 seconds. Type CTRL-C to bypass
Establishing kRemote connection...
initialize physical interface (phy31)
Initializing remote connection...
Sever port = 192.168.1.18:4333, target port = 192.168.1.210:4333

3.1.3 Establishing Ethernet Communications to the DUT

kDiagnostics® enables remote communications as soon as "`kremote.run`" is successfully configured and executed. Choose an Ethernet connection by using the appropriate "$kremote.netdev" value described in section 3.1. No special setup is required for the host computer when an Ethernet is used.

To quickly verify the Ethernet connection to the DUT, use a host ping command with the DUT IP address as the argument. This is the IP address stored in the kDiagnostics® variable "$ip.target".

3.2 Establishing an Optional SLIP Communications to the DUT

Optionally, choose a SLIP connection by using the appropriate "$kremote.netdev" value described in Section 3.1. When using a SLIP connection, SLIP must also be enabled on the host computer.

3.2.1 Creating a SLIP Connection under Windows XP

This section describes how to create a new SLIP connection on the host server computer running Microsoft Windows. This connection is optional and these steps are not required if an Ethernet connection is chosen from the host computer to the DUT.

3.2.2 SLIP — Creating a New Network Connection

Complete the steps below once, for setting up a host computer to communicate with a DUT over SLIP for the first time.

3.2.3 SLIP — Create a New Direct Connection

- Select "***Start->Settings->Control Panel***"
- Double-Click "***Network Connections***"
- Select "***File->New Connection…***"
- Select "***Next***" on the **New Connection Wizard** dialog box.
- Select "***Set up an advanced connection***" and then select "***Next***"
- Select "***Connect directly to another computer***" and then select "***Next***"
- Select "***Guest***" and then select "***Next***"
- Type "**Kozio Target**" in the "***Computer Name***" text field and then select "***Next***"
- Choose the correct COM port in the "***Select a device:***" pull-down selection box and then select "***Next***"
- Select the "***Add a shortcut to this connection to my desktop***" check box.
- Select "***Finish***"

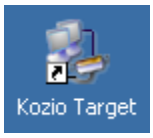
3.2.4 SLIP — Setting the Kozio Target Properties

- The "**Connect Kozio Target**" dialog should appear immediately following step 1.
- Select the "***Properties***" button
- On the "***General***" tab, click "***Configure…***"
- Set the "***Maximum Speed (bps):***" pull-down to **115200** (or as documented in the release notes for the kDiagnostics® firmware image)
- Deselect "***Enable hardware flow control***".
- Ensures that "***Enable modem error control***" and "***Enable modem compression***" are not selected
- Select "***OK***"
- Select the "***Options***" tab
- Deselect the "***Prompt for name and password, certificate, etc.***" check box
- Select the "***Networking***" tab
- Select "***SLIP: Unix Connection***" in the "***Type of dial-up server I am calling:***" pull-down box
- Deselect all options under "***This connection uses the following items:***" except for "***Internet Protocol (TCP/IP)***" and "***QoS Packet Scheduler***".
- Highlight the "***Internet Protocol (TCP/IP)***" option under "***This connection uses the following items:***"
- Select the "***Properties'*** button
- Type in the server IP address to use in the "***Use the following IP address:***" field. This value must match the setting configured in the DUTs "$ip.peer" script value. Both the configured IP addresses "$ip.peer" and "$ip.target" should reside on the same subnet. The DUT command "kremote.debug" displays the settings of all script values controlling the kDiagnostics® remote connection.
- Select the "***Advanced…***" button
- Select the "***General***" tab if not already selected
- Deselect the "***Use default gateway on remote network***" check box
- Deselect the "***Use IP header compression***" check box
- Select "***OK***" on the "**Advanced TCP/IP Settings**" dialog
- Select "***OK***" on the "**Internet Protocol (TCP/IP) Settings**" dialog
- Select "***OK***" on the "**Kozio Target Properties**" dialog

3.2.5 SLIP — Establishing a "Kozio Target" Network Connection

The first time the prior steps are executed, the host computer will immediately attempt to communicate with the DUT via a serial connection. The DUT should be running and have executed the "`kremote.run`" command. If this is true, a network connection between the host computer and the DUT will be established.

3.2.6 SLIP — Restarting the Windows SLIP Connection



After the connection has been disconnected, for whatever reason, reestablishing the connection can be accomplished using the "Kozio Target" desktop icon.

Double-click on the icon and the host will attempt to connect to the DUT.

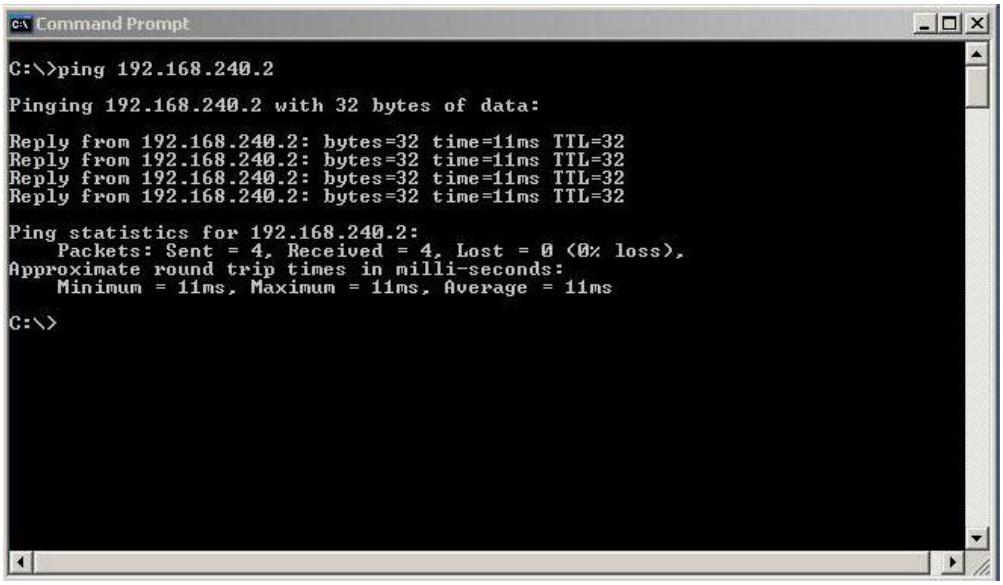
If the connection fails, restart the DUT making sure that the "`kremote.run`" command has been executed. A failed connection displays:

Error Connecting to Kozio Target
Dialing...
Error 777: The connection attempt failed because the modem (or other connecting device) on the remote computer is out of order.
Redial = 60
Cancel
More Info

3.2.7 SLIP — Verifying the Windows SLIP Connection

Follow these instructions to verify the SLIP connection:

- Launch the "**Network Connections**" dialog box if it is not already running.
  - Select "***Start->Settings->Control Panel***"
  - Double-Click on "***Network Connections***"
- Under "***Direct***", the "**Kozio Target**" should have a "***Status***" of "**Connected**" with possibly other attributes.
- As a second step, ping the DUT:
  - Launch a "**Command Prompt**" window
  - Execute "ping 192.168.240.2" but use the DUT's SLIP IP address. This value is a kDiagnostics® remote connection variable as described in Section 3.1kDiagnostics® Remote Communications. If the IP address has been changed, be sure to use the new value.


Command Prompt
C:\>ping 192.168.240.2
Pinging 192.168.240.2 with 32 bytes of data:
Reply from 192.168.240.2: bytes=32 time=11ms TTL=32
Reply from 192.168.240.2: bytes=32 time=11ms TTL=32
Reply from 192.168.240.2: bytes=32 time=11ms TTL=32
Reply from 192.168.240.2: bytes=32 time=11ms TTL=32
Ping statistics for 192.168.240.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 11ms, Maximum = 11ms, Average = 11ms
C:\>

3.3 Kozio Properties Files

KozioServer™ and the client applications each use a properties file for configuration values. These files are located in the root folder of the Kozio Host Tools installation directory, e.g. "C:\Program Files\Kozio". The following files are used:

Application Program	**Properties File**
KozioServer	kozioserver.properties
SequenceRunner and SequenceRunnerCLI	seqrunner.properties
SequenceReporter	kozioreporter.properties
ValidationAssistant	va.properties

NOTE: *kozioserver.properties* contains configuration data linking one or more Virtual Fixtures to their IP Addresses. A Virtual Fixture is either an actual hardware networking device provided by Kozio or the DUT directly. The client applications, such as ValidationAssistant, refer to a Fixture which is mapped by KozioServer to an actual DUT. This allows the client applications to always refer to the same Fixture even though a given DUT may be repeatedly swapped out for a new DUT on the production line.

Verify that this information matches your network configuration.

The lines below are an example of the Fixture to IP address declarations. If this file is altered, the Kozio Server must be restarted.

The installation program will only generate settings for one Fixture. If you are licensed for multiple Fixtures, you must manually edit kozioserver.properties to insert the additional settings (Fixture2 through Fixture4 in this example).

```
NumberOfFixtures  = 4
Fixture1          = 192.168.1.200
Fixture2          = 192.168.1.201
Fixture3          = 192.168.1.202
Fixture4          = 192.168.1.203
```

Starting with Release 4.1, fixture names may optionally be assigned in the properties file as shown in the following example:

```
FixtureName1 = Debby 8313 DUT 1
FixtureName2 = Debby 8313 DUT 2
FixtureName3 = Debby Blaze
```

If fixture names are not specified, the fixtures will be named Fixture1, Fixture2, etc. by default.

Update the properties file any time a change is made to the IP addresses for the server or device under test. By specifying these values during installation, changes to this file will not be needed unless the IP address(es) of the DUT(s) changes. If *kozioserver.properties* is modified, KozioServer™ must be stopped and started. Refer to Section 3.3.3 for details.

3.3.1 KozioServer Properties File Variables

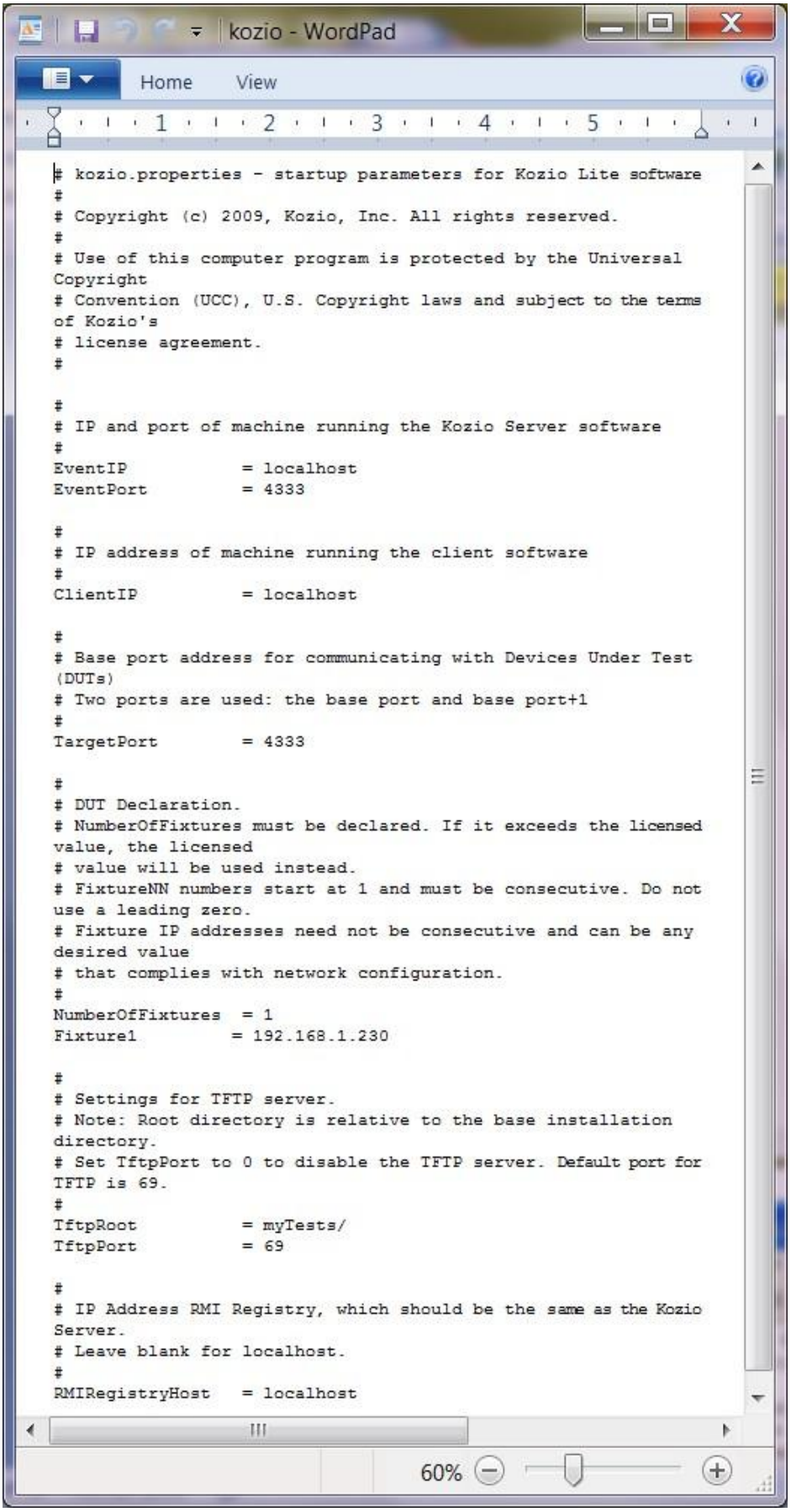
This table describes the variables found in *kozioserver.properties*. Modify the variables as needed.

Variable	Example	Description	Notes
EventIP	192.168.1.32	The IP address of the computer running KozioServer™.	
EventPort	4333	The base IP port used by KozioServer™. Two port numbers are used: EventPort and EventPort+1.	The port number for the host computer(s) and kDiagnostics® remote communications variable $kremote.port must match (section 3.1)
ClientIP	192.168.1.32	The IP address of the computer running a client application such as ValidationAssistant™.	This value is only used by the client application and has no affect on the server settings.
TargetIP	192.168.1.100	The IP address of the DUT.	This value must match the kDiagnostics® remote communications variable $ip.target (Section 3.1).
TargetPort	4333	The IP port used by the by the Device Under Test.	This value must match EventPort.
TftpRoot	myTests/	This variable specifies the root folder where files downloaded from the host to the DUT will be stored.	This is the place to store kScript files, image files for downloading to the DUT, and test sequence files.
TftpPort	69	The TFTP port number.	Set this value to zero (0) to disable the TFTP server.
RMIRegistryHost	192.168.1.32	The IP address of the computer running KozioServer™.	

3.3.2 SequenceRunner Properties File Variables

This table describes the variables found in *seqrunner.properties*. Modify the variables as needed.

	Example	Description	Notes
LogToFile	true	Valid options are “true” or “false”.	Set to “true” to log all test results to text file stored in the “/logs” folder under the main installation folder.
LogFileName	koziolog	The name of the file used for test logging when LogToFile is “true”.	Created in the “/logs” folder under the main installation folder.
LogFileExtension	.log	The extension of the file used for test logging when LogToFile is “true”.	
ReportFileName	koziореport	The name of the file used for test reports.	
ReportFileExtension	.log	The extension of the file used for test reports.	
LogFileSave	always	Valid options are “always”, “never”, or “error”.	This variable determines when a detailed execution log is saved to disk. This log is useful for troubleshooting.
DBURL	jdbc:postgresql://192.168.1.22/kozio	Valid JDBC URL for the database.	This field is only needed if the optional database is installed and enabled.
DBUserName	kozio_user	User name for database access.	Used by application programs to access the database.
DBPassword	koziouser	Password for database access	




```
# kozio.properties - startup parameters for Kozio Lite software
#
# Copyright (c) 2009, Kozio, Inc. All rights reserved.
#
# Use of this computer program is protected by the Universal
Copyright
# Convention (UCC), U.S. Copyright laws and subject to the terms
of Kozio's
# license agreement.
#

#
# IP and port of machine running the Kozio Server software
#
EventIP           = localhost
EventPort         = 4333

#
# IP address of machine running the client software
#
ClientIP          = localhost

#
# Base port address for communicating with Devices Under Test
(DUTs)
# Two ports are used: the base port and base port+1
#
TargetPort        = 4333

#
# DUT Declaration.
# NumberOfFixtures must be declared. If it exceeds the licensed
value, the licensed
# value will be used instead.
# FixtureNN numbers start at 1 and must be consecutive. Do not
use a leading zero.
# Fixture IP addresses need not be consecutive and can be any
desired value
# that complies with network configuration.
#
NumberOfFixtures  = 1
Fixture1         = 192.168.1.230

#
# Settings for TFTP server.
# Note: Root directory is relative to the base installation
directory.
# Set TftpPort to 0 to disable the TFTP server. Default port for
TFTP is 69.
#
TftpRoot          = myTests/
TftpPort          = 69

#
# IP Address RMI Registry, which should be the same as the Kozio
Server.
# Leave blank for localhost.
#
RMIRegistryHost   = localhost
```

3.3.3 Restarting KozioServer™

Anytime `kozioserver.properties` is modified, KozioServer™ must be stopped and restarted.

NOTE: Using "***Restart***" does not work. The proper sequence is to first "***Stop*****"** then "***Start*****"** KozioServer™.

1. From the Windows task bar, select "***Start***"
2. Select ***"Control Panel***" (for Vista users, it is recommended to switch to 'Classic' mode to execute these instructions).
3. Select "***Administrative Tools***"
4. Select "***Services***"
5. Highlight the line labeled "***KozioServer***"
6. Select "***Stop***" the service.
7. Select "***Start***" the service.

NOTE: It is good practice to run ValidationAssistant™ and type in the simple "`version`" command to verify server to DUT communications.

4
ValidationAssistant™

4.1 Getting Started

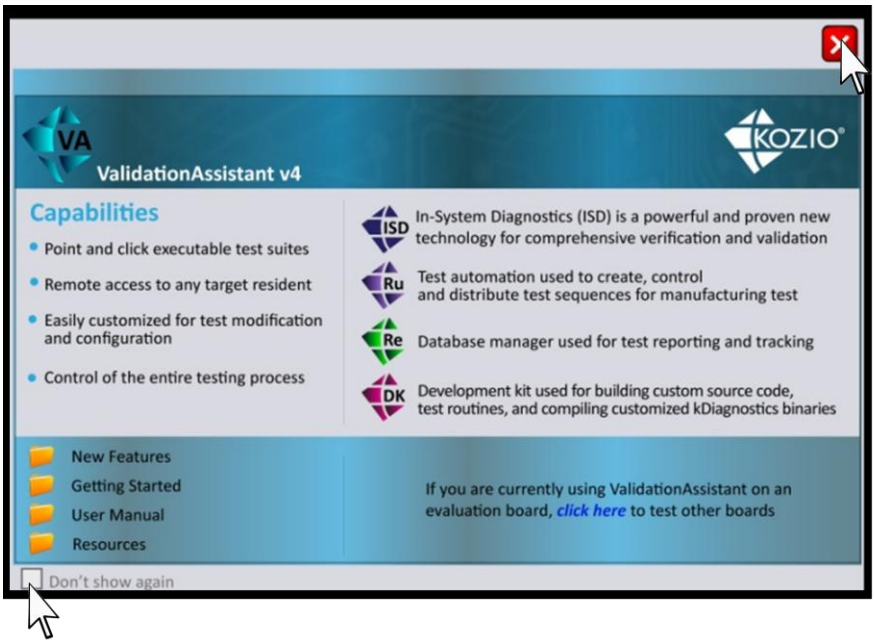
ValidationAssistant™ provides a means to execute test commands and acts as an event listener displaying all information received from a device under test (DUT). Launch ValidationAssistant™ using the desktop shortcut.

4.1.1 Start-Up Banner

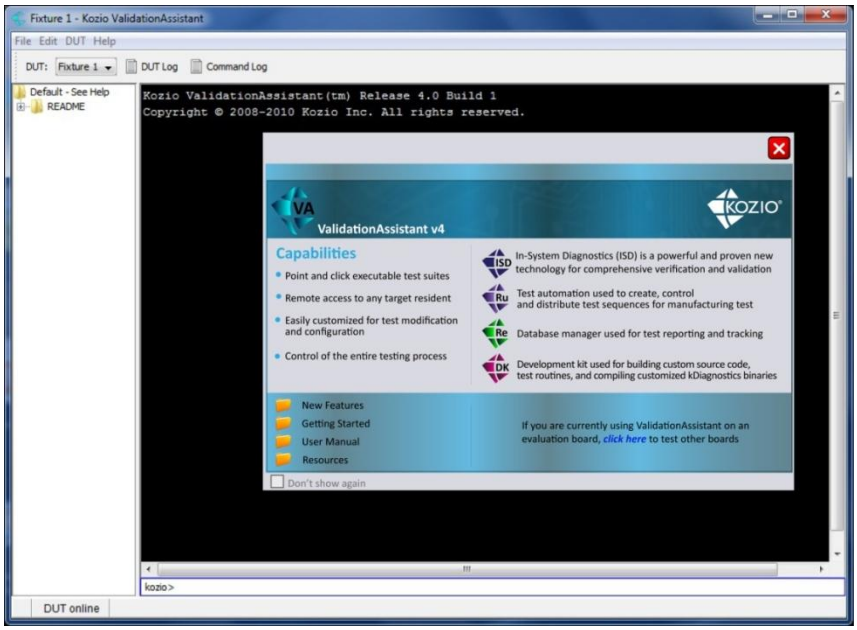
ValidationAssistant™ displays a startup banner followed by a welcome screen and the main operator interface.

4.1.2 Welcome Screen

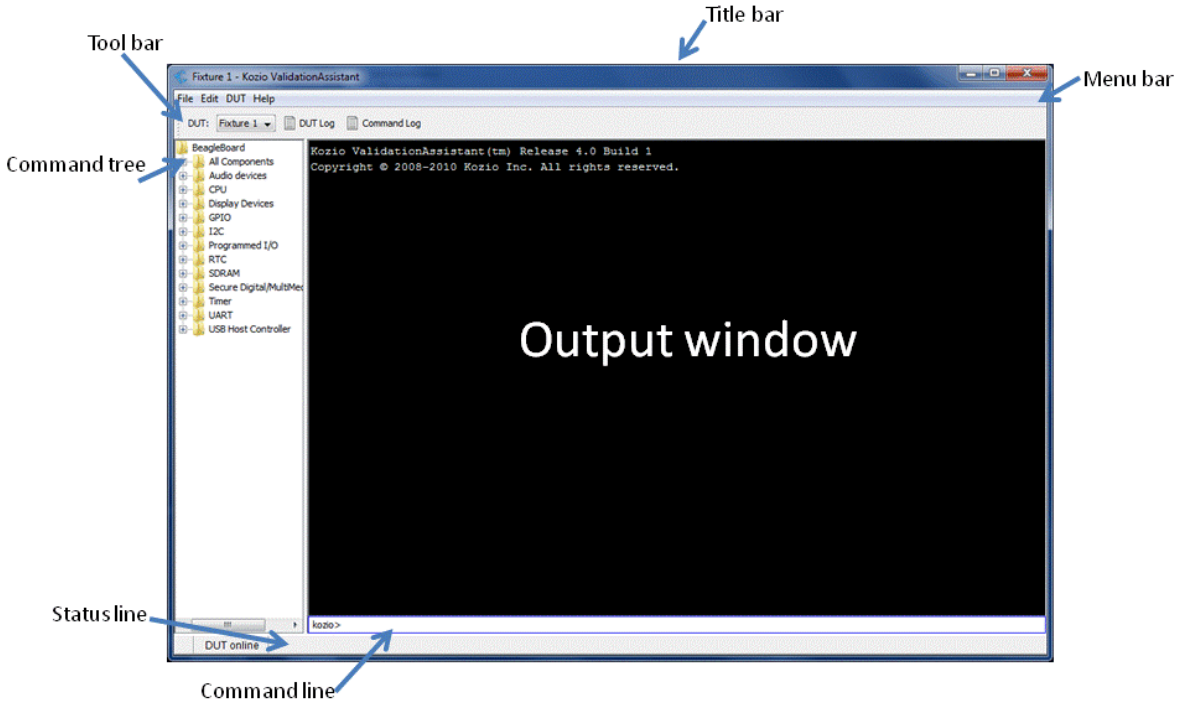
A welcome screen will pop-up providing shortcuts to valuable information and documents. Click the red "***X***" in the upper right-hand corner of the screen to close. A "***Don't show again***" box is also provided in the bottom left-hand corner.


ValidationAssistant v4
KOZIO
Capabilities
Point and click executable test suites
Remote access to any target resident
Easily customized for test modification and configuration
Control of the entire testing process
In-System Diagnostics (ISD) is a powerful and proven new technology for comprehensive verification and validation
Test automation used to create, control and distribute test sequences for manufacturing test
Database manager used for test reporting and tracking
Development kit used for building custom source code, test routines, and compiling customized kDiagnostics binaries
New Features
Getting Started
User Manual
Resources
If you are currently using ValidationAssistant on an evaluation board, click here to test other boards
Don't show again

Use ValidationAssistant™ to execute test commands on the DUT as specified by 'Fixture'. (If multiple devices are being tested simultaneously, see Chapter 5 for more information on multi-device testing.)

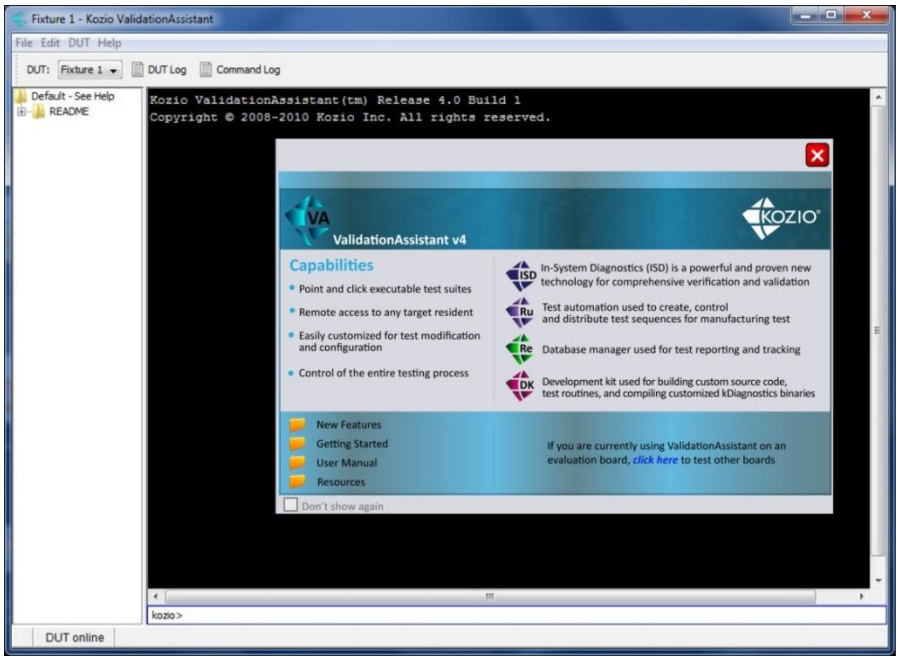

Fixture 1 - Kozio ValidationAssistant
File Edit DUT Help
DUT: Fixture 1
DUT Log
Command Log
Default - See Help
README
Kozio ValidationAssistant(tm) Release 4.0 Build 1
Copyright © 2008-2010 Kozio Inc. All rights reserved.
ValidationAssistant v4
KOZIO
Capabilities
Point and click executable test suites
Remote access to any target resident
Easily customized for test modification and configuration
Control of the entire testing process
In-System Diagnostics (ISD) is a powerful and proven new technology for comprehensive verification and validation
Test automation used to create, control and distribute test sequences for manufacturing test
Database manager used for test reporting and tracking
Development kit used for building custom source code, test routines, and compiling customized kDiagnostics binaries
New Features
Getting Started
User Manual
Resources
If you are currently using ValidationAssistant on an evaluation board, click here to test other boards
Don't show again
kozio>
DUT online

4.1.3 Screen Overview



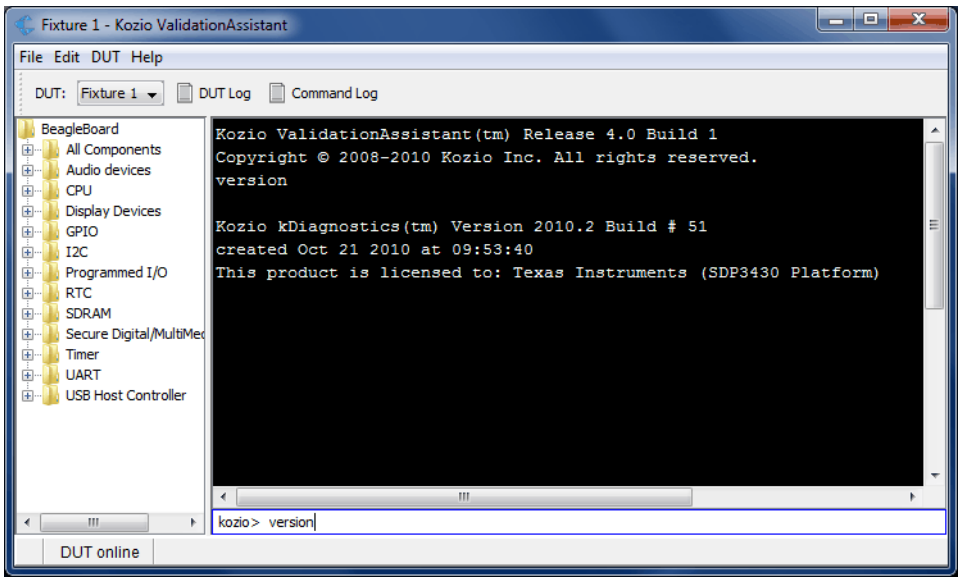
4.2 DUT Status

Located at the bottom on the "***Status Line***" is the DUT indicator showing whether the DUT is online or offline. Online indicates that a status command was successfully sent from the server to the DUT and back again. Offline indicates a communications problem between the server and DUT.


Fixture 1 - Kozio ValidationAssistant
File Edit DUT Help
DUT: Fixture 1
DUT Log
Command Log
Default - See Help
README
Kozio ValidationAssistant(tm) Release 4.0 Build 1
Copyright © 2008-2010 Kozio Inc. All rights reserved.
ValidationAssistant v4
KOZIO
Capabilities
Point and click executable test suites
Remote access to any target resident
Easily customized for test modification and configuration
Control of the entire testing process
In-System Diagnostics (ISD) is a powerful and proven new technology for comprehensive verification and validation
Test automation used to create, control and distribute test sequences for manufacturing test
Database manager used for test reporting and tracking
Development kit used for building custom source code, test routines, and compiling customized kDiagnostics binaries
New Features
Getting Started
User Manual
Resources
If you are currently using ValidationAssistant on an evaluation board, click here to test other boards
Don't show again
kozio>
DUT online

4.3 Validating Setup

Run ValidationAssistant™ and type in "***version***" in the command line to verify quickly that all steps have been completed properly.

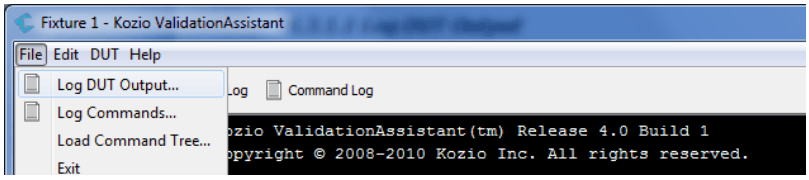


4.4 Menu Bar

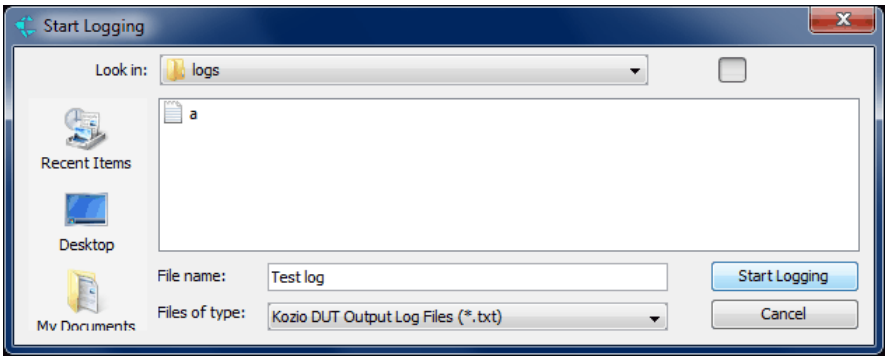
4.4.1 File Menu

4.4.1.1 Log DUT Output

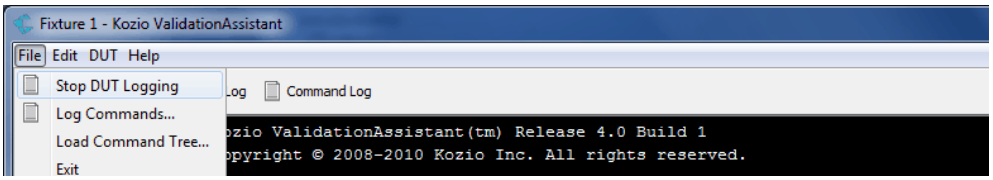
On the Menu bar, click on "***File***" and then on "***Log DUT Output***".



This command will bring up a pop-up window. Type in a file name for the DUT log and click on the "***Start Logging***" button. This will begin the logging process which logs all interactions with the DUT This is useful for sharing information with support or other engineers.



A Stop DUT logging command becomes available in the "***File***" menu once the logging command has been executed.

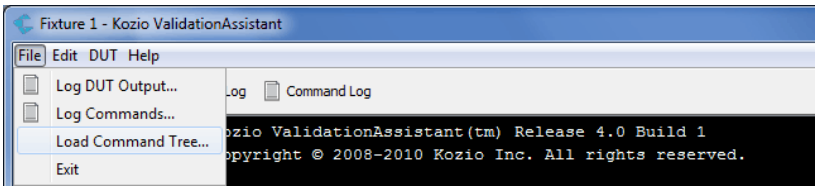


4.4.1.2 Log Commands

Command logging is also available in ValidationAssistant. Click on the "***Log Command***" menu from the "***File***" menu to execute this command. This feature logs all commands sent to the target and logs only commands. It is useful for capturing commands and then creating a script file of the specific commands you need to execute in order to complete a given task.

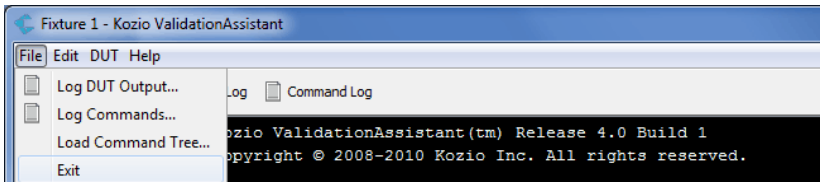
4.4.1.3 Load Command Tree

See Section 4.6.1 for more information on loading the command tree.



4.4.1.4 Exit

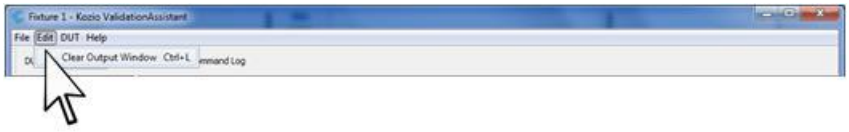
Go to the "***File***" menu located on the menu bar and click on "***Exit***" to close ValidationAssistant™.



4.4.2 Edit Menu

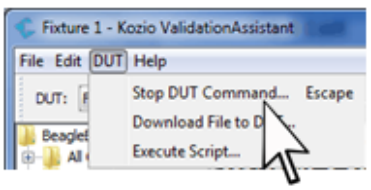
4.4.2.1 Clear Window

Go to the "***File***" menu and click. This will bring the pull down menu up. Click on the "***Clear Output Window***" option to clear all of the text previously displayed text in the results box.



4.4.3 DUT Menu

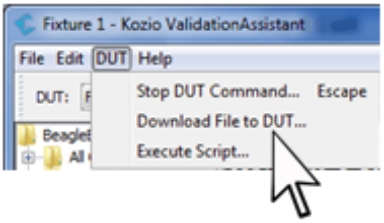
4.4.3.1 Stop DUT Command



Use the "***Stop DUT Command"*** option to cancel a command in progress. You can also use the "Escape" key as a shortcut for aborting a command in progress.

4.4.3.2 Download File to DUT

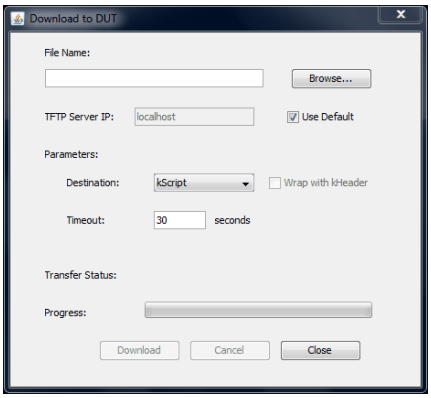
To download files to the DUT, select the DUT item located on the menu bar.



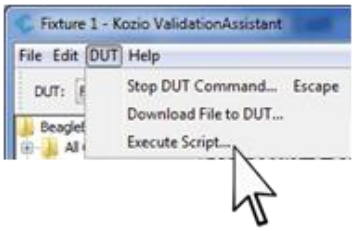
This will bring up a pop-up menu. Select the menu option "***Download File to DUT…***" which will bring up a dialog box. Type in the name of the file to be downloaded, or click on the "***Browse***" radio button to locate the file. Click on the "***Download***" radio button to begin the operation.

Note: For this release, to program a device on the target with a downloaded file the user must select an image file that has been wrapped using the Kozio utility kHeader.exe. This utility is freely available from Kozio. This utility will encode information into the image file such as where to

program the data on the target, such as Flash memory. This method also provides a cyclic redundancy check (CRC) for the image to verify the transfer.



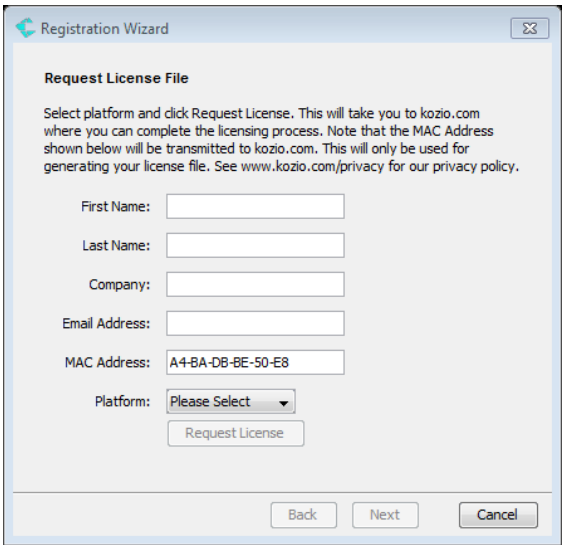
4.4.3.3 Execute Script

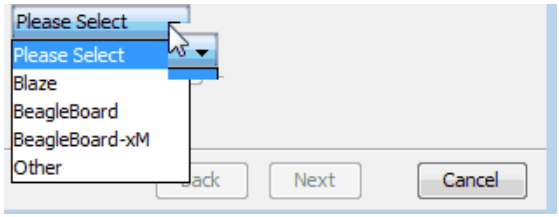


The execute script command can be found on the "***DUT***" menu located on the menu bar. Click on the "***Execute Script***" option to download and execute a script.

4.4.4 Help Menu

Product registration can be found under the "***Help***" command which is located on the menu bar. To register, click on "***Register Product***" under the "***Help***" menu and fill out the form with the requested information. From the ***Platform*** pull-down, select the platform being used for the DUT. Listed on the ***Platform*** pull-down are evaluation platforms and a separate item for purchased products. These two choices work by taking you to a Kozio web page form where you data is transferred. A license file will be generated and sent to the email address included in the registration information.

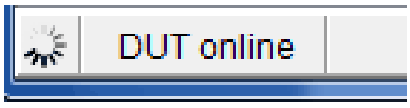

Registration Wizard
Request License File
Select platform and click Request License. This will take you to kozio.com where you can complete the licensing process. Note that the MAC Address shown below will be transmitted to kozio.com. This will only be used for generating your license file. See www.kozio.com/privacy for our privacy policy.
First Name:
Last Name:
Company:
Email Address:
MAC Address:
A4-BA-DB-BE-50-E8
Platform:
Please Select
Request License
Back
Next
Cancel



4.5 Status Line Indicators

A green status bar located on the "Status Line**" at the botton of ValidationAssistant** indicates the status of the test being run.

A spinning arrow at the bottom left-hand side of the screen indicates the DUT is busy executing a command. This indicator only appears when the DUT is busy executing a command.



The DUT online or offline status is located on the status line next to the busy indicator.

4.6 Commands

The kDiagnostics architecture is built using a hierarchy of commands that range from low-level commands for probing the hardware to fully functional test suites that can test every component on the board.

4.6.1 Executing Commands

Use ValidationAssistant™ to execute test commands on the DUT as specified by the 'DUT' selector on the task bar. (See Chapter 5 for more information if multiple devices are under test simultaneously.)

ValidationAssistant provides two ways to execute Commands. Commands can be run through the **Command Tree** (see section 4.3) or the **Command Line** (see section 4.11).

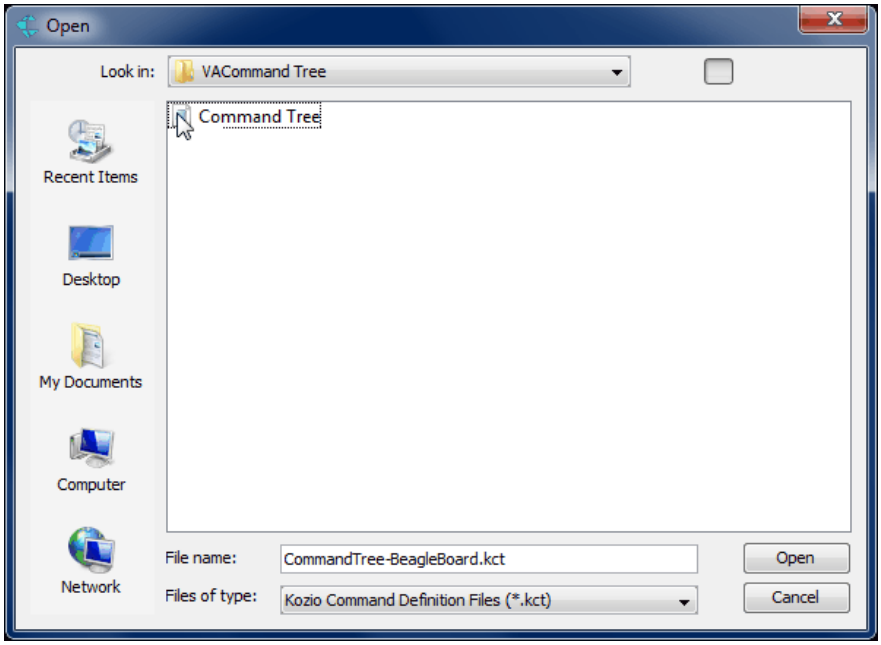
4.7 Command Tree

The command tree can be found on the left side of the screen and is organized by the hardware components available on a platform. The command tree is customized for each platform that kDiagnostics is delivered on by providing commands only for those components available on that platform. Under each component, a second level of commands is divided into five potential groups, but all groups are not available for all components. The groups available are: Suites, Diagnostics tests, Performance tests, Test methods, and Low-level commands. These groups are defined in great detail in subsequent sections.

4.7.1 Load Command Tree

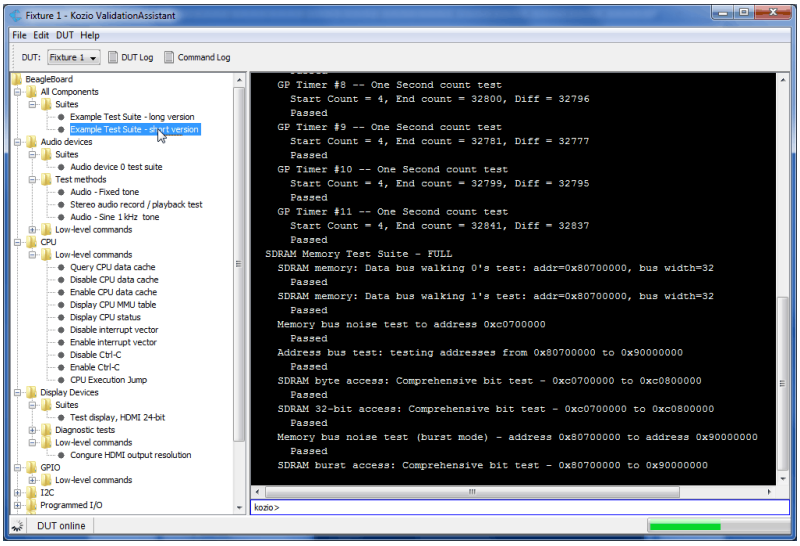
To load the command tree, use the "***File"*** menu option and click on "***Load Command Tree"*** from the top menu options.

Find the location of the ****.kct*** file (Kozio Command Tree File) for the DUT and double click. The command tree will automatically load into ValidationAssistant™.



4.7.2 Executing Tests through the Command Tree

Navigate down to the specific component to be run and double click or use the "***Execute Script***" option located in the "**Menu Bar**" to select a script file that is sent to the DUT for interpretation.
Commands can also be executed by typing the command in the **Command Line**. (See Section 4.7 for more information on the **Command Line**.


Fixture 1 - Kozio ValidationAssistant
File Edit DUT Help
DUT: Fixture 1
DUT Log
Command Log
BeagleBoard
All Components
Suites
Example Test Suite - long version
Example Test Suite - short version
Audio devices
Suites
Audio device 0 test suite
Test methods
Audio - Fixed tone
Stereo audio record / playback test
Audio - Sine 1 kHz tone
Low-level commands
CPU
Low-level commands
Query CPU data cache
Disable CPU data cache
Enable CPU data cache
Display CPU MMU table
Display CPU status
Disable interrupt vector
Enable interrupt vector
Disable Ctrl-C
Enable Ctrl-C
CPU Execution Jump
Display Devices
Suites
Test display, HDMI 24-bit
Diagnostic tests
Low-level commands
Congure HDMI output resolution
GPIO
Low-level commands
I2C
Programmed I/O
GP Timer #8 -- One Second count test
Start Count = 4, End count = 32800, Diff = 32796
Passed
GP Timer #9 -- One Second count test
Start Count = 4, End count = 32781, Diff = 32777
Passed
GP Timer #10 -- One Second count test
Start Count = 4, End count = 32799, Diff = 32795
Passed
GP Timer #11 -- One Second count test
Start Count = 4, End count = 32841, Diff = 32837
Passed
SDRAM Memory Test Suite - FULL
SDRAM memory: Data bus walking 0's test: addr=0x80700000, bus width=32
Passed
SDRAM memory: Data bus walking 1's test: addr=0x80700000, bus width=32
Passed
Memory bus noise test to address 0xc0700000
Passed
Address bus test: testing addresses from 0x80700000 to 0x90000000
Passed
SDRAM byte access: Comprehensive bit test - 0xc0700000 to 0xc0800000
Passed
SDRAM 32-bit access: Comprehensive bit test - 0xc0700000 to 0xc0800000
Passed
Memory bus noise test (burst mode) - address 0x80700000 to address 0x90000000
Passed
SDRAM burst access: Comprehensive bit test - 0x80700000 to 0x90000000
kozio>
DUT online

4.7.3 Command Tree Prompt

For a number of test commands, kDiagnostics requires confirmation from the operator. When ValidationAssistant receives such a prompt from the target, a dialog box is presented to the operator. Click "**Yes** or "**No**" to complete each action. Selecting "**No**" will fail the test.

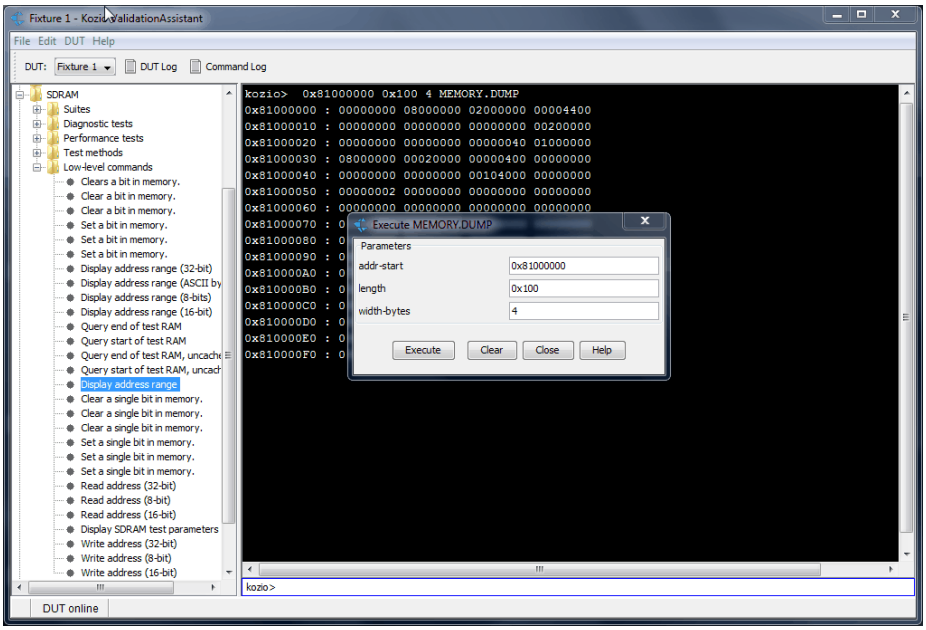
4.8 Command Line

Tests can be executed through the **Command Line.** Type in the name of the command in the **Command Line** and press the ***Enter*** key. ValidationAssistant™ will send each command to the DUT and all output from kDiagnostics will appear in the **Output Window**.
To send multiple commands to the the DUT, select the text and copy it, then select the **Command Line** and paste the text. All copies lines will be sent to the DUT for interpretation by kDiagnostics and all output will be sent to the **Output Window**.
This option can also be used to send an entire script file to the DUT.
A third option of executing commands stored in a script file is to browse for the file using a tool such Windows Explorer and then drag the file onto the Output Window. All lines of the file will be sent to the DUT and interpreted by kDiagnostics.

4.9 Low-Level Commands

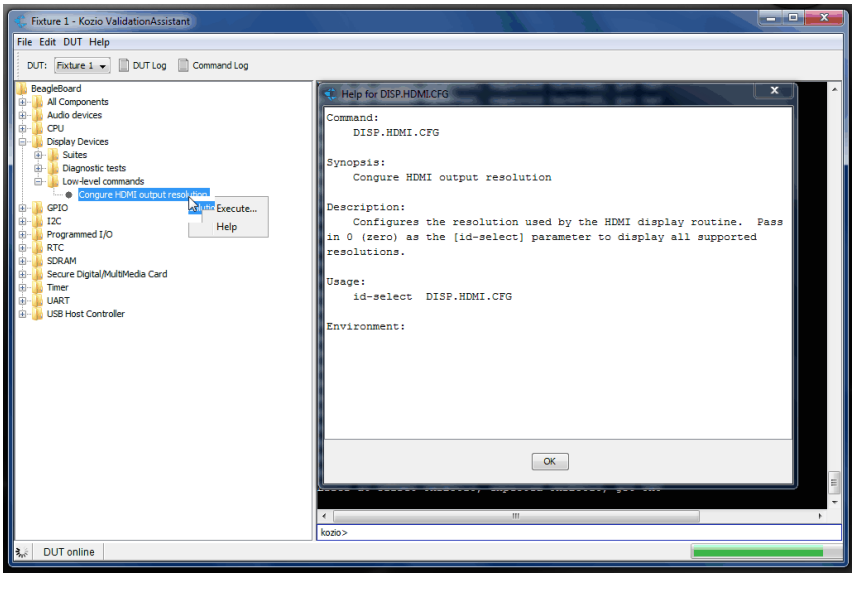
Many low-level commands are available for the operator to execute. Low-level commands require parameters. Each high-level component, such as SDRAM, may have low-level command. Double click on the command to be executed. A dialogue box will appear prompting for the parameters. Type in the parameters for the specific command and click "**Execute**".

A right-mouse button pop-up menu is available for all Low-level Commands. One menu option displays help for the command and one executes the command.



4.9.1 Low-Level Help

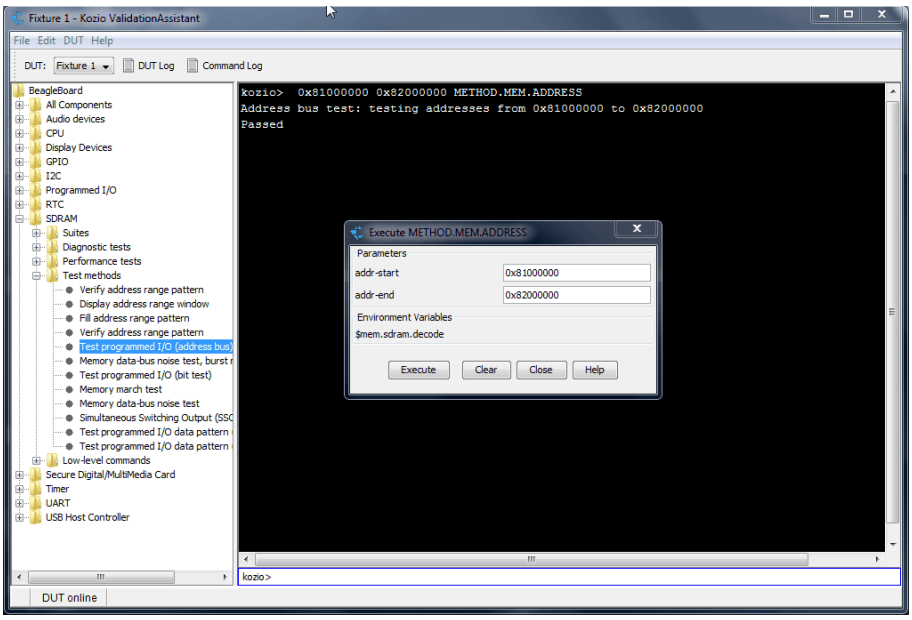
A help menu is available for all low-level commands. To access this menu, select the command and then use the right mouse button. This will bring up the "***Help***" option. Click on the "**Help"** button. A box will appear on the screen providing ***command***, ***synopsis***, ***description***, ***usage*** and ***environment*** for the selected command.



4.10 Test Methods

Test methods are a set of diagnostics tests that deliver a pass / fail result. Double click on the name of a test method to run it. Type in the parameters for the specific command and click on "**Execute**". The results will be displayed in the **Output Window**.

A right-mouse button pop-up menu is available for all Test Methods. One menu option displays help for the command and one executes the command.



4.11 Performance Tests

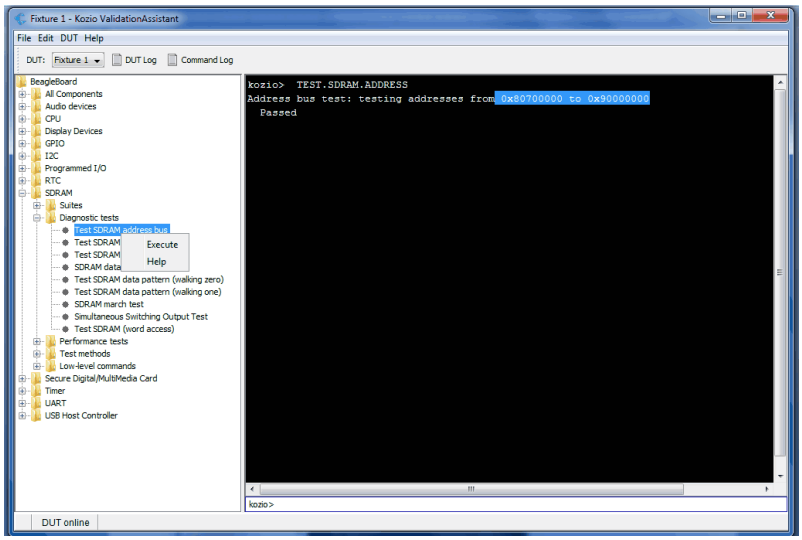
Some components, such as SDRAM, have a number of performance tests associated with that component. Double-click on a performance test to display a characterization of a given performance characteristic.

A right-mouse button pop-up menu is available for all Performance Tests. One menu option displays help for the command and one executes the command.

4.12 Diagnostic Tests

A set of diagnostic tests has been built on top of test methods. The tests are pre-configured for a custom board based on the layout and build of materials for that board design. The diagnostic tests deliver a pass/fail result and do not require parameters. For instance, if the SDRAM address bus test is run, it will use an SDRAM Address Test to test the entire range of SDRAM memory that is available on the platform.

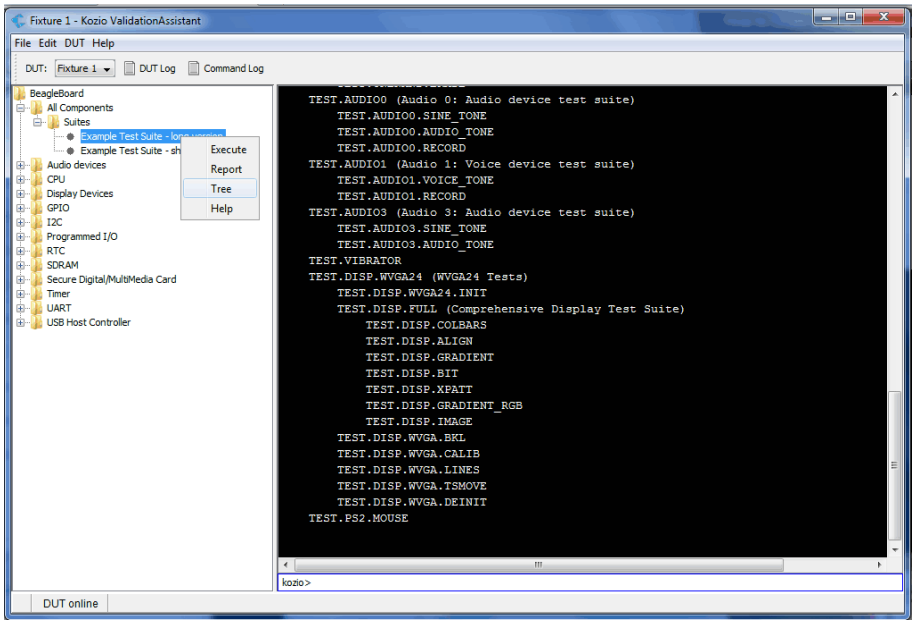
A right-mouse button pop-up menu is available for all Diagnostic Tests. One menu option displays help for the command and one executes the command.



4.13 Test Suites

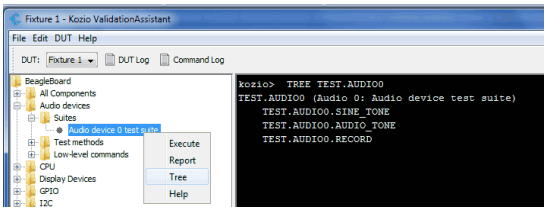
Test suites are collections of diagnostic tests or other test suites. Quick and comprehensive test suites are available for all devices. Test suites can also be nested. For example, under the "**All Components**" folder on the command tree, over 140 tests are packaged into a single command labeled "**Example Test Suite – long version**". Multiple components are tested and there are multiple levels of nesting under each suite.

A right-mouse button pop-up menu is available for all Test Suites. One menu option displays help for a test and one executes the test. Test Suites also have two additional menu options labeled "Tree" and "Report" which are described in the next two sections.



4.13.1 Test Suites Tree

The Tree command can be accessed by right clicking on a highlighted test suite. Select "**Tree**" and a complete listings of all of the nested suites and tests that are included in the suite will be displayed on the **Output Window**.

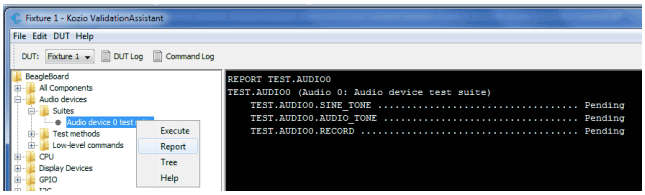


This same information can be accessed by entering the "tree" command followed by a test suite name on the command line. For example, on the command line enter "tree test.sdram" and press the Enter key.

4.13.2 Test Suites Report

Click on "**Report**" and ValidationAssistant displays a full report on the selected test suite. The information displayed is the test name followed by a status of Pending, Passed, or Failed. Pending indicates that the test has not been executed yet. This information is cleared when the platform is rebooted.

This same information can be accessed by entering the "report" command followed by a test suite name on the command line. For example, on the command line enter "report test.sdram" and press the Enter key.

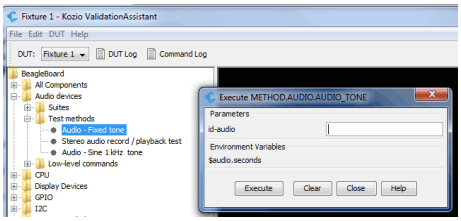


4.14 Troubleshooting Test Failures

In the case of test failures, ValidationAssistant provides several different options for troubleshooting.

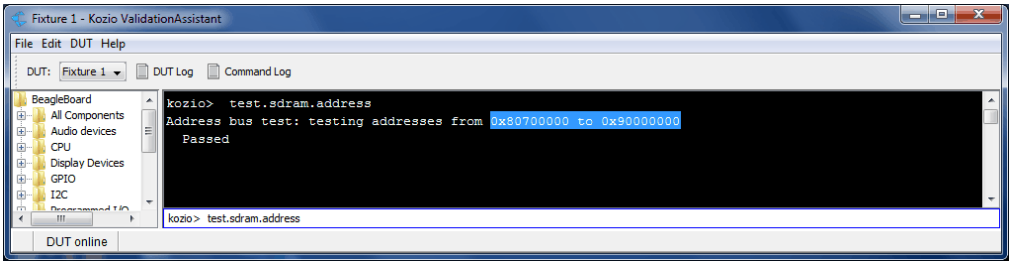
4.14.1 Troubleshooting with Test Methods

Go to the ***Test Methods*** command and change the input parameters down to a smaller range or a specific device that is failing (see section 4.9 for details on Test Methods). A logic analyzer or other hardware devices can be hooked up to try to diagnose the failure.



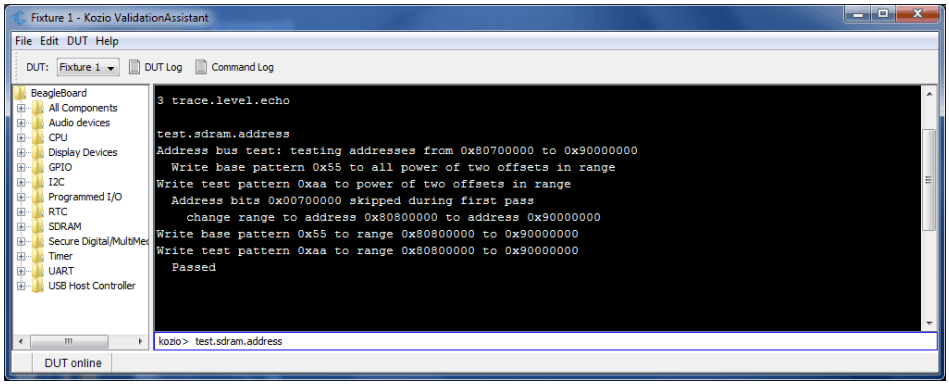
4.14.2 Troubleshooting with Tracing

kDiagnostics has tracing built into the product and this can be accessed through ValidationAssistant. For example, if the command ***test.sdram.address*** is typed into the command line, limited information is provided. In this case, only the test range information is displayed in the ***Output Window*** for testing the address bus bits.

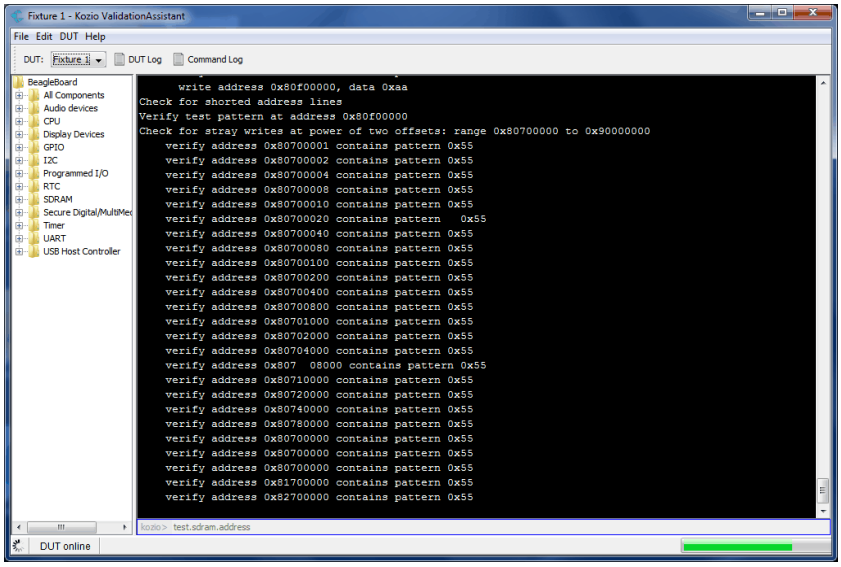


To change the trace level, type the trace level in the **Command Line** and hit "***Enter***." The command to change the trace level echoing is "<level> trace.level.echo" where <level> is an integer from 0 to 6. The value of zero turns off all echoing of trace commands and the larger the integer the more data that is displayed on the **Output Window**.

After changing the trace echo level, reenter the command ***test.sdram.address***. More information about the test being run is displayed about what the test is doing in the **Output Window**. For example, with a trace level of 3, a few additional lines of information is displayed.



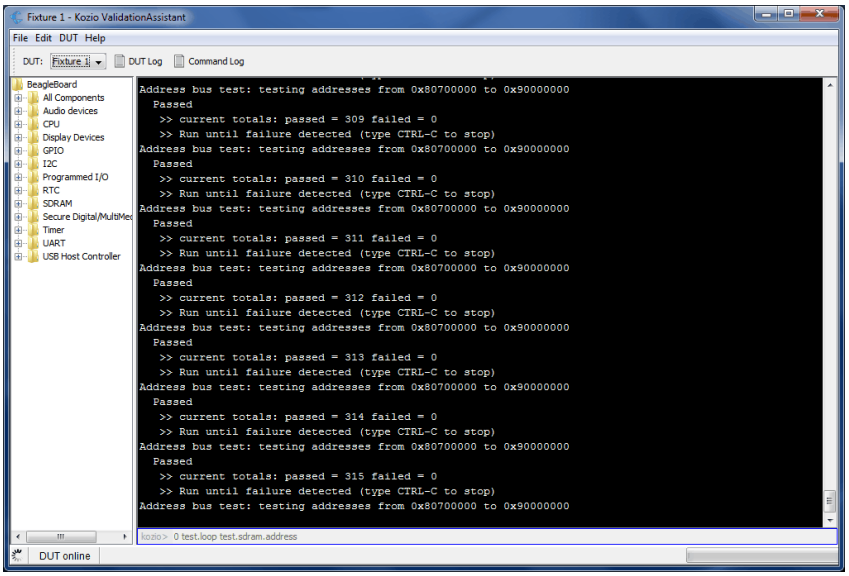
If the trace level is turned up to 6, every single read and write memory access is displayed while executing this test. The ***tracing*** feature provides additional information that is very useful for fault isolation and learning how to configure interfaces and devices..

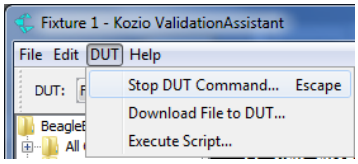

Fixture 1 - Kozio ValidationAssistant
File Edit DUT Help
DUT: Fixture 1 DUT Log Command Log
BeagleBoard
All Components
Audio devices
CPU
Display Devices
GPIO
I2C
Programmed I/O
RTC
SDRAM
Secure Digital/MultiMed
Timer
UART
USB Host Controller
write address 0x80f00000, data 0xaa
Check for shorted address lines
Verify test pattern at address 0x80f00000
Check for stray writes at power of two offsets: range 0x80700000 to 0x90000000
verify address 0x80700001 contains pattern 0x55
verify address 0x80700002 contains pattern 0x55
verify address 0x80700004 contains pattern 0x55
verify address 0x80700008 contains pattern 0x55
verify address 0x80700010 contains pattern 0x55
verify address 0x80700020 contains pattern 0x55
verify address 0x80700040 contains pattern 0x55
verify address 0x80700080 contains pattern 0x55
verify address 0x80700100 contains pattern 0x55
verify address 0x80700200 contains pattern 0x55
verify address 0x80700400 contains pattern 0x55
verify address 0x80700800 contains pattern 0x55
verify address 0x80701000 contains pattern 0x55
verify address 0x80702000 contains pattern 0x55
verify address 0x80704000 contains pattern 0x55
verify address 0x807 08000 contains pattern 0x55
verify address 0x80710000 contains pattern 0x55
verify address 0x80720000 contains pattern 0x55
verify address 0x80740000 contains pattern 0x55
verify address 0x80780000 contains pattern 0x55
verify address 0x80700000 contains pattern 0x55
verify address 0x80700000 contains pattern 0x55
verify address 0x80700000 contains pattern 0x55
verify address 0x81700000 contains pattern 0x55
verify address 0x82700000 contains pattern 0x55
kozio> test.sdram.address
DUT online

4.14.3 Troubleshooting with Looping

The looping feature provides a way to troubleshoot problems or run regression suites on the board. In the **Command Line,** type in the number of times the test is to loop followed by the command, "**<loop_count> test.loop test.sdram.address**" and hit "***Enter***", where <loop_count> is the number of times to execute the command. Use a loop_count of zero (0) to run infinite times. To stop the looping, use the "**Esc**" key. If a failure is detected, the test will stop automatically. This feature can be used for to loop any command.

There are alternate ways of looping available and those are described in the "kDiagnostics kScript Reference" document.


Fixture 1 - Kozio ValidationAssistant
File Edit DUT Help
DUT: Fixture 1 DUT Log Command Log
BeagleBoard
All Components
Audio devices
CPU
Display Devices
GPIO
I2C
Programmed I/O
RTC
SDRAM
Secure Digital/MultiMed
Timer
UART
USB Host Controller
Address bus test: testing addresses from 0x80700000 to 0x90000000
Passed
>> current totals: passed = 309 failed = 0
>> Run until failure detected (type CTRL-C to stop)
Address bus test: testing addresses from 0x80700000 to 0x90000000
Passed
>> current totals: passed = 310 failed = 0
>> Run until failure detected (type CTRL-C to stop)
Address bus test: testing addresses from 0x80700000 to 0x90000000
Passed
>> current totals: passed = 311 failed = 0
>> Run until failure detected (type CTRL-C to stop)
Address bus test: testing addresses from 0x80700000 to 0x90000000
Passed
>> current totals: passed = 312 failed = 0
>> Run until failure detected (type CTRL-C to stop)
Address bus test: testing addresses from 0x80700000 to 0x90000000
Passed
>> current totals: passed = 313 failed = 0
>> Run until failure detected (type CTRL-C to stop)
Address bus test: testing addresses from 0x80700000 to 0x90000000
Passed
>> current totals: passed = 314 failed = 0
>> Run until failure detected (type CTRL-C to stop)
Address bus test: testing addresses from 0x80700000 to 0x90000000
Passed
>> current totals: passed = 315 failed = 0
>> Run until failure detected (type CTRL-C to stop)
Address bus test: testing addresses from 0x80700000 to 0x90000000
kozio> 0 test.loop test.sdram.address
DUT online

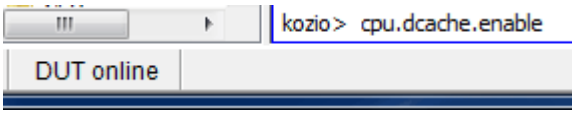


To stop a running test, go to the "***DUT***" menu located on the menu bar and click on the "***Stop DUT Command*** " or press the "***Esc***" key on the keyboard.

4.14.4 Memory Test

It is important to note that if you interrupt a test in progress kDiagnostics may leave data caching turned off. If data caching is disabled, all commands will run much slower.

To display the current status of data caching, enter the command "cpu.dcache .hex". This will display a 0x0 or 0x1 indicating that data caching is either off or on, respectively.

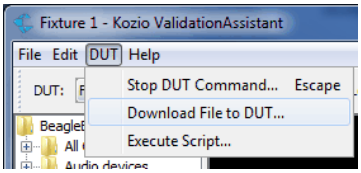


To turn the data cache back on, enter the command "**cpu.dcache.enable**" and press "***Enter***". This will turn data caching back on.

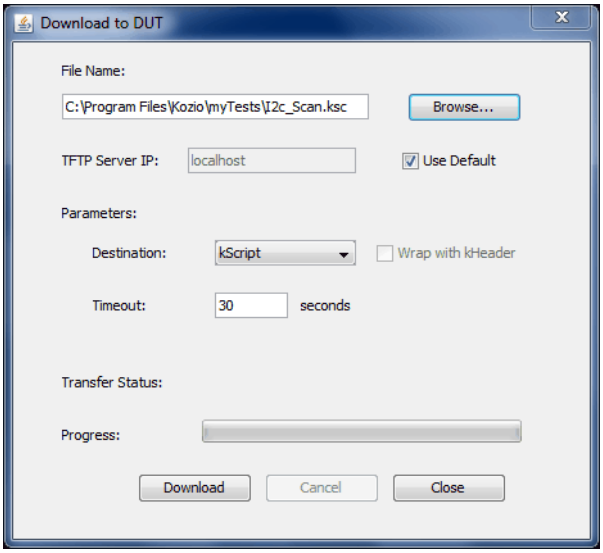
A full list of all available commands is provided in the "kDiagnostics Command Reference". This document is provided with purchased products only.

4.15 Downloading Scripts

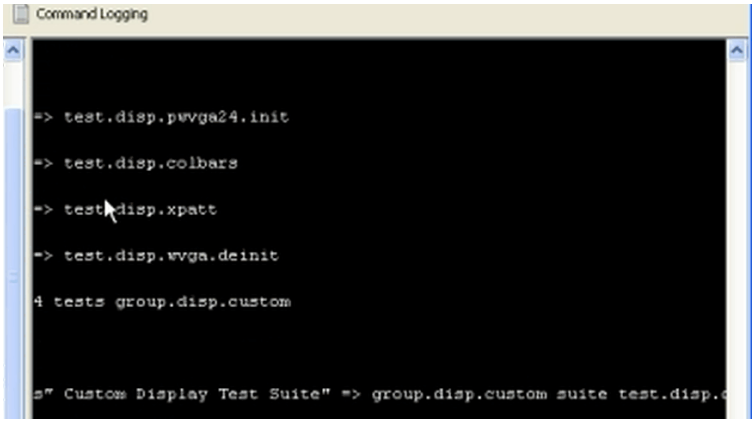
To download a script file select the "***DUT***" menu and then select "***Download File to DUT.***"



This will bring up a pop-up menu. Click on the "**Browse**" button and find the location of the desired *.ksc file. Kozio uses the *.ksc file extension to differentiate Kozio kScript files. Select the file. Then click on the "***Download***" button located in the pop-up window.



The scripts will automatically be downloaded to the DUT and the also displayed in the **Output Window**.



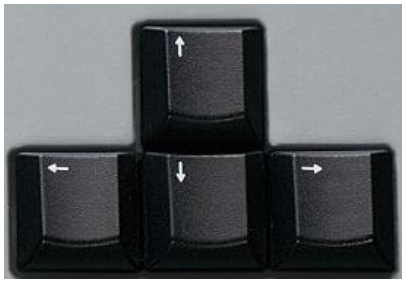
For more information on Script and Command creation, see the ***kDiagnostics User Manual***.

4.16 TFTP File Transfer

TFTP file transfer is available through ValidationAssistant for kScript and kHeader-wrapped files only. The kHeader program is freely distributed by Kozio. Please inquire through support@kozio.com if you would like to receive this additional tool.

4.17 Shortcuts, Keyboard

To do this	Press
Clear the screen	CTL + L
Abort a command	Esc
Cut	CTL + X
Copy	CTL + C
Paste	CTL + V



ValidationAssistant provides a buffer of previously executed commands. This includes all commands typed in as well as executed through the command tree.

To re-enter a previous command, press the **up arrow** or down-arrow key on the keyboard. This will cycle through all commands entered. Press the ‘Enter” key to execute that command. You can also scroll commands and then modify the command by selecting the Command Line text entry box and making modifications before pressing Enter.

5
SequenceRunner™

5.1 Launching SequenceRunner™

Launch SequenceRunner™ using the desktop shortcut.

SequenceRunner™ displays a startup banner followed by the main operator interface.

Figure 5.1. SequenceRunner™ – Startup Banner

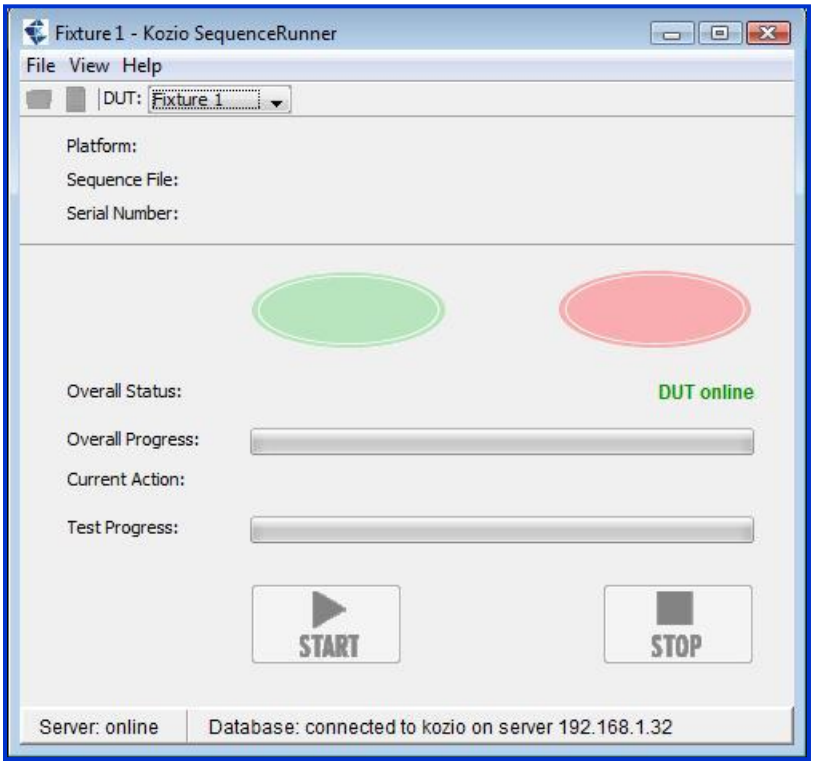


Figure 5.2. SequenceRunner™ – Initial Screen with DUT Online

5.2 Log In

The operator must log in. Select the "File" menu and then select the "Login…" option as shown below.

Enter the operator's name. This name is stored in the central database as the operator's name and is tracked with all test executions. Once logged in, additional menu options are available to open a test sequence file and to logout.

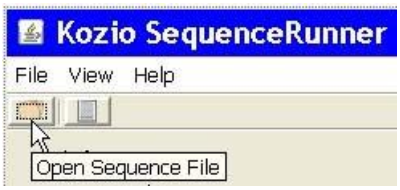
To switch operators, select the "File->Logout" menu option and then log in the new operator.

5.3 Executing Test Sequences

One of the main purposes of SequenceRunner™ is to execute test sequences. All test sequences are stored in ASCII files using XML tags. Find additional information on the format of test sequence files in Section 5.13 Test Sequence Files. Several example sequence files and kScript files are provided under the "examples" folder in `C:/Program Files/Kozio/examples`. To start the testing process, load a test sequence file using one of two methods:

- Choose "***File -> Open Sequence…***" and then select the ***Open Sequence File*** icon.

- Choose the task bar icon for Open Sequence File



The **Open** dialog box allows browsing of the test sequence files available. Select the desired test sequence file and then select ***Open***. SequenceRunner™ opens and validates the file. Section 5.12 XML Validation section describes the validation process.

NOTE: Anytime a test sequence file is modified, that file must be reopened so that it is re-validated.

After the test sequence file is loaded, the title bar, "Platform:" text, and "Sequence File:" text are updated based on data from the test sequence file. In addition, the ***START*** button becomes active.

NOTE: Sample test sequence files have been provided, but these must be checked and edited, if necessary, to match the DUT.

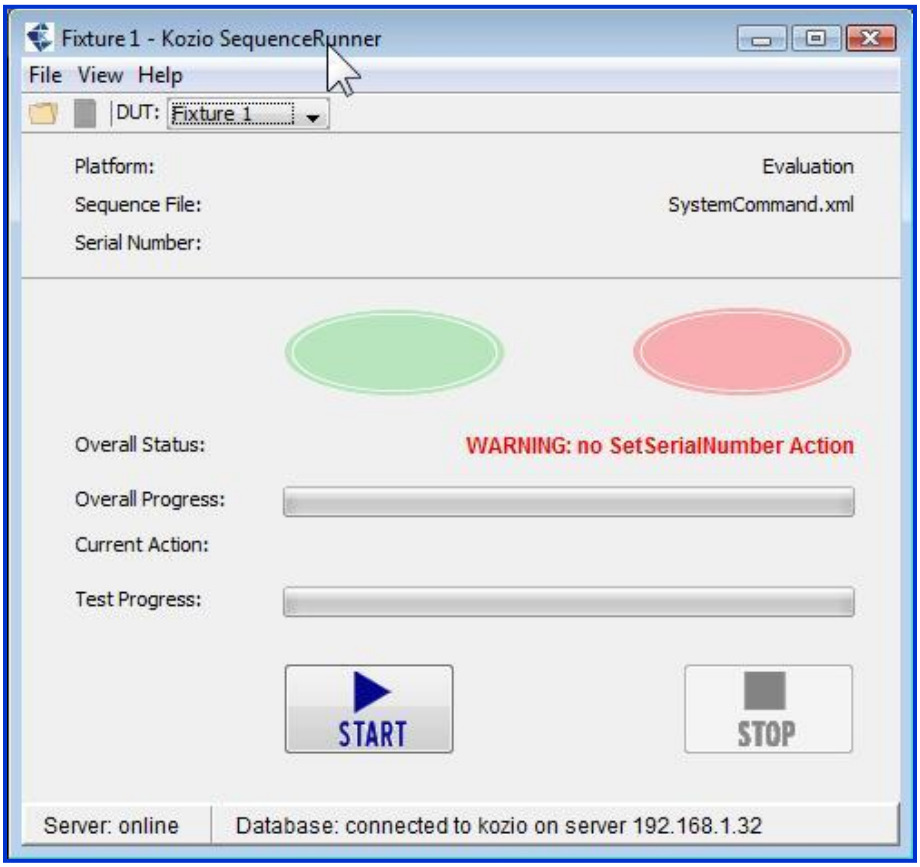


Figure 5.3. SequenceRunner™ – With SystemCommand.xml Loaded

Press the ***START*** button to begin executing a test sequence. As the test sequence executes the screen will be updated as follows:

- ***DUT:*** This specification selects the fixture under test. Default Fixture Number = 1.
- ***Serial Number***: This field is updated by the SetSerialNumber action.
- ***Overall Status:*** This field is updated with a pass or fail message after the test sequence is executed.
- ***Overall Progress:*** This progress bar is updated after each action in the test sequence is completed.
- ***Current Action:*** This field is updated as each action is executed. The text displayed is taken directly from the "Name" attribute of each action tag in the XML file.
- ***Test Progress:*** This field displays status information directly from the DUT. The test designer does not have control over this text.

Select the "**Start**" button to execute the test sequence.

Here is an example of a test in progress:

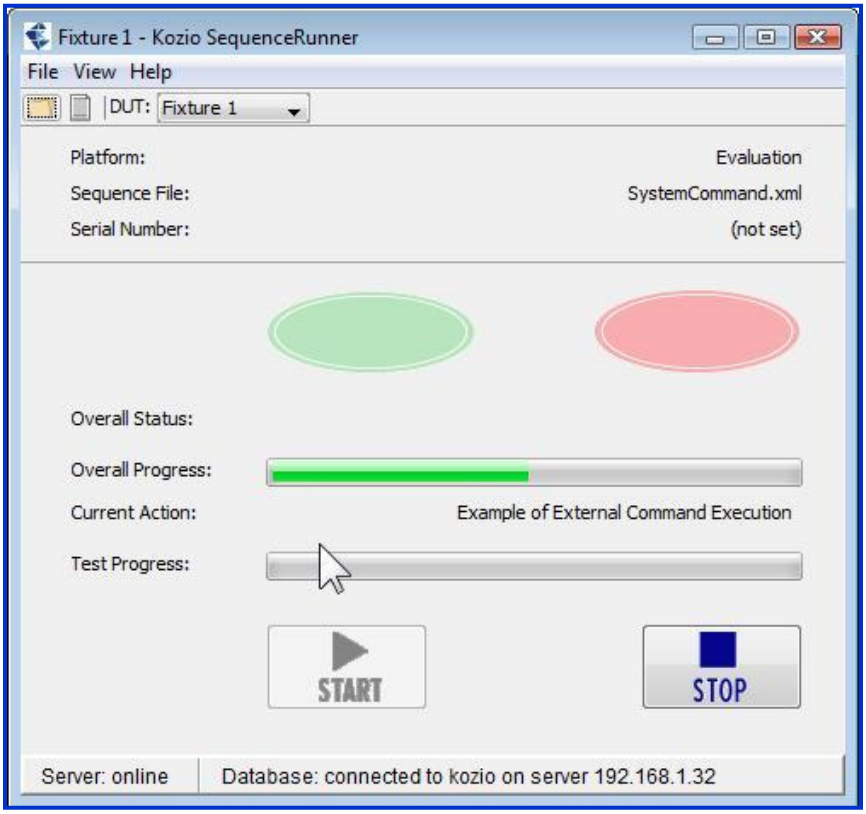


Figure 5.4. SequenceRunner™ – Test in Progress

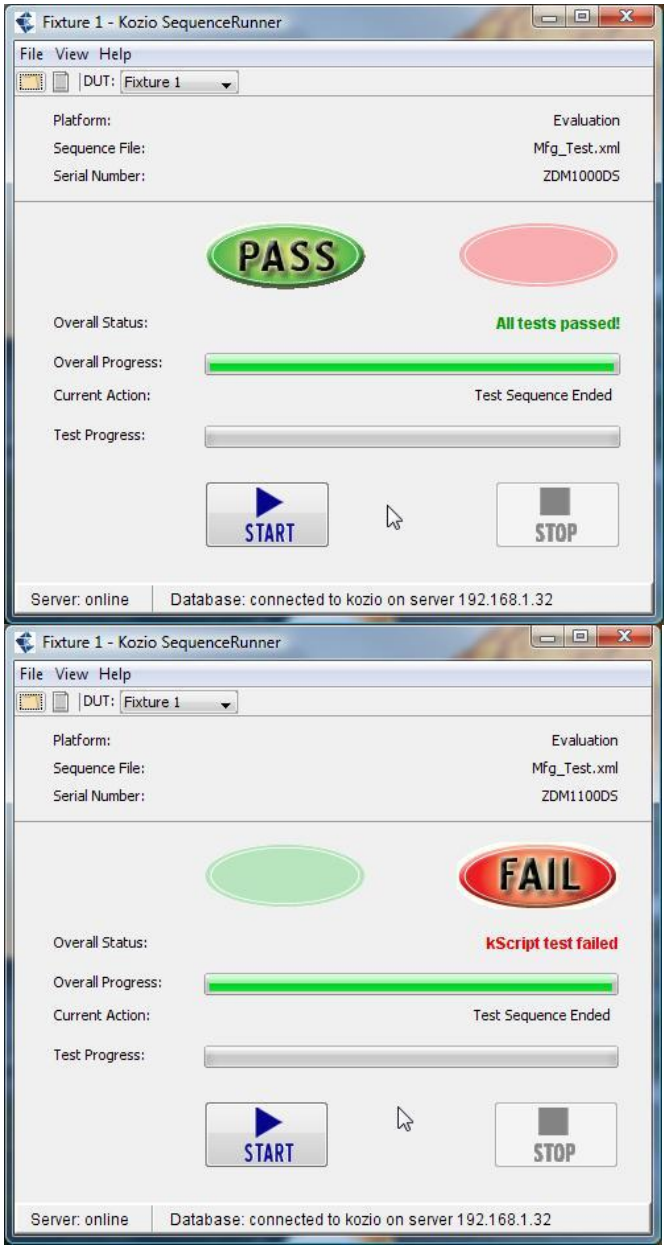


Figure 5.5. SequenceRunner™ – Test PASS & FAIL Status

5.4 Detailed Log

A SequenceRunner™ log file stores a record of all activities. This log file is useful for debugging a configuration or setup issue. View the log file using one of the following two methods:

- Using "***View -> Detailed Log***"
- Alternately, view the log file using SequenceReporter™ and the integrated database. For more information see Chapter 6 SequenceReporter™.

An example of a run log is shown below after executing a version command on the DUT.

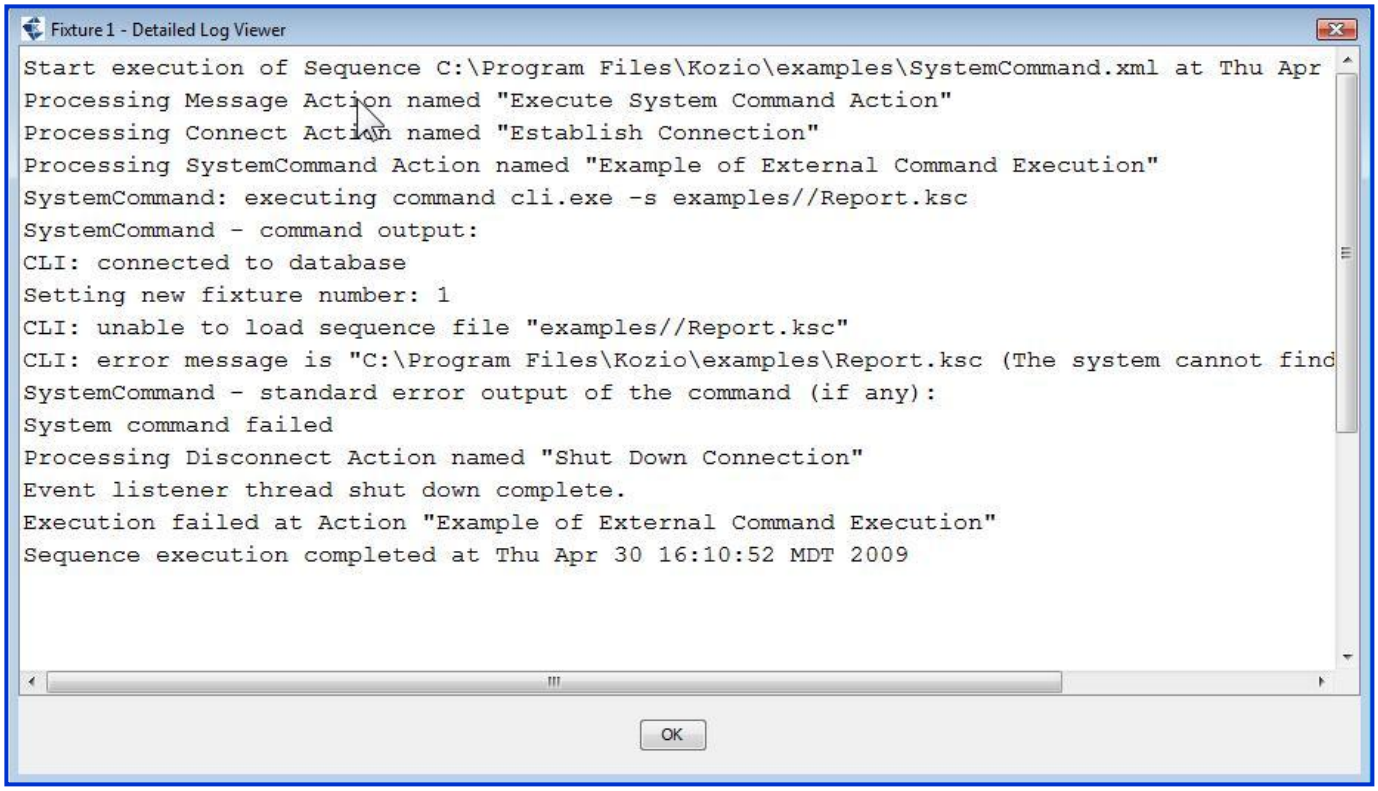

Fixture 1 - Detailed Log Viewer
Start execution of Sequence C:\Program Files\Kozio\examples\SystemCommand.xml at Thu Apr
Processing Message Action named "Execute System Command Action"
Processing Connect Action named "Establish Connection"
Processing SystemCommand Action named "Example of External Command Execution"
SystemCommand: executing command cli.exe -s examples//Report.ksc
SystemCommand - command output:
CLI: connected to database
Setting new fixture number: 1
CLI: unable to load sequence file "examples//Report.ksc"
CLI: error message is "C:\Program Files\Kozio\examples\Report.ksc (The system cannot find
SystemCommand - standard error output of the command (if any):
System command failed
Processing Disconnect Action named "Shut Down Connection"
Event listener thread shut down complete.
Execution failed at Action "Example of External Command Execution"
Sequence execution completed at Thu Apr 30 16:10:52 MDT 2009
OK

Figure 5.6. SequenceRunner™ – Detailed Log Viewer

5.5 Command Line Interface (CLI)

The Command Line Interface (CLI) allows test commands and script files to be sent to kDiagnostics® running on the device under test (DUT). Select the fixture under test using the -f option. The CLI program communicates with KozioServer which then sends commands to the DUT specified by the -f option. The default value for -f is 1. The CLI can be invoked directly from a command window or from third party programs, so it can easily be added into existing production test processes.

The CLI is a host tool that can be used by third party programs to execute Kozio functional tests on the DUT. The CLI tool can be called by scripting languages such as Perl. The main benefit is the ability to quickly add Kozio functional tests into an existing production test process.

5.6 CLI Return Value

In accordance with DOS program conventions, the CLI returns a positive integer exit status code. A return value of zero (0) indicates success. A positive value indicates that an error occurred, as shown in the following table:

Exit Status Code	Meaning
0	Success
1	kDiagnostics test failed or sequence execution failed

2	Input error, e.g. file not found
3	Runtime error, e.g. no license file
4	Invalid kDiagnostics command or other DUT error

The best way to capture the exact error codes for each failure is to redirect output to a log file and parse that log file when a positive return value is encountered. More information about the failure may also be found in the Kozio log file, *logs\koziologs.log*.

5.7 CLI Options

Execute "sequencerunner`cli -h`" to view a list of available options:

```
usage: SequenceRunnerCLI [options] command
 -abort                 Abort currently executing command
 -f <fixture-number>    Specify Fixture Number
 -h                     Print help for this application
 -help                  Print help for this application
 -i <ip-address>        Specify DUT IP address
 -k <script-name>       Execute kScript file
 -l <log-file>          Capture output to logfile
 -o                     Overwrite old logfile contents
 -q                     Execute commands in quiet mode
 -r <repeat-count>      Repeat command
 -s <sequence-name>     Load and execute sequence file
 -t <cmd-timeout>       Specify command timeout value (seconds)
 -u <user-name>         Set user name
 -v                     Display version information
 -version               Display version information
```

5.8 Executing Commands with CLI

To execute a test command use:

sequencerunner`cli` *`command`*

The "*`command`*" string is any valid kDiagnostics® command such as "`test.sdram`", "version", "`cpu.mmu`", etc. This option sends the command to the DUT for immediate interpretation. Results are displayed to the output console and the overall success of execution is given by the program's exit status code. A list of available commands is documented in the *kDiagnostics® Command Reference* manual.

5.9 Executing a Test Script with CLI

To execute a test script for a single fixture use:

sequencerunner`cli -k` *`script-file-name`*

NOTE: CLI defaults to Fixture 1 when the "-f" parameter is not specified. To execute a test script for a Fixture other than the default, use:

sequencerunnercli –f # -k *script-file-name*

Where # is the desired Fixture number and *script-file-name* is any valid kDiagnostics® script file (*.ksc).

This sends the entire script file to the DUT for line-by-line interpretation. Results are displayed to the output console and the overall success of execution is returned as a DOS exit status code as shown in the table above [TODO: need table number].

5.10 Executing a Test Sequence with CLI

Kozio Test Sequence files may be executed from CLI with the –s option as shown here:

sequencerunnercli –s *sequence-file-name*

where *sequence-file-name* is the name of any valid Kozio Test Sequence file, for example "mytests\mfg_test.xml". The –f option may also be used to specify the desired Fixture number, as with any CLI command.

Report Files

A report file for each test suite execution is created based on the configuration options chosen in `seqrunner.properties`. The report filename includes the Serial Number that is retrieved from a ScanBarcode action. All report files are stored in the "logs" directory and in the database. This report can be printed and packaged with a DUT.

View the report file using one of three methods:

- Using "***View -> Summary Report***"

- Using the View Summary Report button on task bar.

- Alternately, the test report can be accessed using SequenceReporter™, if an integrated database is being used. This is discussed in Chapter 6 SequenceReporter™.

A sample summary report file is shown below:

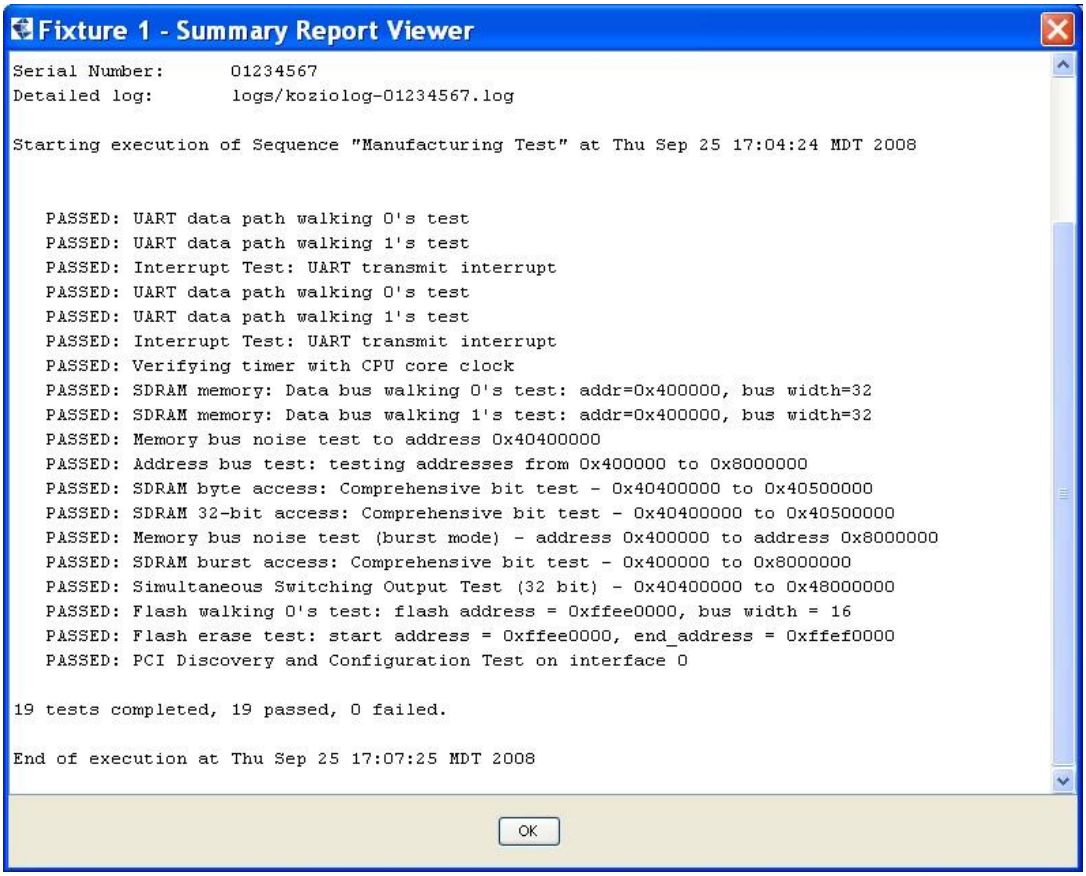

Fixture 1 - Summary Report Viewer
Serial Number: 01234567
Detailed log: logs/koziolog-01234567.log
Starting execution of Sequence "Manufacturing Test" at Thu Sep 25 17:04:24 MDT 2008
PASSED: UART data path walking 0's test
PASSED: UART data path walking 1's test
PASSED: Interrupt Test: UART transmit interrupt
PASSED: UART data path walking 0's test
PASSED: UART data path walking 1's test
PASSED: Interrupt Test: UART transmit interrupt
PASSED: Verifying timer with CPU core clock
PASSED: SDRAM memory: Data bus walking 0's test: addr=0x400000, bus width=32
PASSED: SDRAM memory: Data bus walking 1's test: addr=0x400000, bus width=32
PASSED: Memory bus noise test to address 0x40400000
PASSED: Address bus test: testing addresses from 0x400000 to 0x8000000
PASSED: SDRAM byte access: Comprehensive bit test - 0x40400000 to 0x40500000
PASSED: SDRAM 32-bit access: Comprehensive bit test - 0x40400000 to 0x40500000
PASSED: Memory bus noise test (burst mode) - address 0x400000 to address 0x8000000
PASSED: SDRAM burst access: Comprehensive bit test - 0x400000 to 0x8000000
PASSED: Simultaneous Switching Output Test (32 bit) - 0x40400000 to 0x48000000
PASSED: Flash walking 0's test: flash address = 0xffee0000, bus width = 16
PASSED: Flash erase test: start address = 0xffee0000, end_address = 0xffef0000
PASSED: PCI Discovery and Configuration Test on interface 0
19 tests completed, 19 passed, 0 failed.
End of execution at Thu Sep 25 17:07:25 MDT 2008
OK

Figure 5.7. SequenceRunner™ – Sample Summary Test Report File

5.11 SQL Database

The database provides a means for tracking test stations, operators, and test results. A full test report is stored for each DUT based on barcode and serial number.

The DUT serial number can be a simple barcode label value, or a database mapping of barcode values to UUT Serial Number can be created. The database must be pre-filled with Barcode/UUT Serial Number mappings for this method.

The entire database schema is open. The pgAdmin tool can also be used to traverse the database and execute queries. A graphical reporting tool called SequenceReporter™ provides a quick and easy means of creating custom reports.

5.12 XML Validation

Every test sequence file must follow a strict set of XML rules defined in a file called KozioSequenceFile.dtd. Every time a new test sequence file is opened in SequenceRunner™, it is validated against this set of rules. The syntax rules of XML are very simple and very strict.

5.13 Test Sequence Files

Test sequence files are ASCII text files containing XML tags describing the circuit board test process. Each XML performs a given test action. Create test sequence files that combine various test actions required for the test process.

The first line in a test sequence file is the XML declaration and it defines the XML version and the character encoding used in the document. Such as:

```
<?xml version="1.0" ?>
```

All sequence file documents must conform to the 1.0 specification of XML. For more information on the specification, see: http://www.w3.org/TR/REC-xml/.

The next line declares the root element of the document and points to the Document Type Definition (DTD) file:

```
<!DOCTYPE TestSequence SYSTEM "KozioSequenceFile.dtd">
```

The first tag is the <TestSequence> tag. Within enclosing TestSequence tags, one or more action tags can be listed. Action tags are defined in section 5.13.2 Action Tags.

Some quick notes:

- All XML elements must have a closing tag
- XML tags are case sensitive
- With XML, the tag <Letter> is different from the tag <letter>
- Opening and closing tags must therefore be written with the same case
- XML elements must be properly nested
- XML attribute values must be quoted
- With XML, White Space is preserved
- The syntax for writing comments in XML is similar to that of HTML
  - `<!—This is a comment →`
- With XML, it is illegal to omit the closing tag

5.13.1 TestSequence Tag

Defines a test sequence with a varying number of action tags. <TestSequence> … </TestSequence>

Attributes

Author The author's name specified will appear in the database and in the report files.

SequenceName This name appears in the database and in the report file.

Platform The name of the platform that will appear in SequenceRunner™, the database, and report files.

Example

```
<TestSequence  Author="Kozio" SequenceName="Example" Platform="Evaluation">
```

5.13.2 Action Tags

Test sequences are created using actions that allow the operator to be prompted, connection to a DUT to be established, execution of test commands, and a number of other options. A test sequence file contains any number of action tags. Here is listing of available actions.

Action Tag	Brief Description
Connect	Establish a connection with a DUT.
Disconnect	Disconnect from a DUT.
DownloadImage	Download a binary file from the host to the DUT and store it in the indicated memory location.
DownloadScript	Download a kScript file to the DUT using a running TFTP server. This is the most efficient means of transferring an entire script file for later execution.
ExecuteCommand	Execute a test kScript command on the DUT.
GetDatabase	Retrieve a value from a database. The user chooses which table to access.
GetFile	Get the contents of a disk file and store it in a variable.
GetValue	Parse the text stored in a variable and return a substring.
Message	Display a message to the operator.
OperatorComment	Prompt the operator to enter a comment.
Prompt	Prompt the operator for input data.
PutDatabase	Store a value in the database. The user chooses which table to update.
PutFile	Save sequence variable contents to disk file.
ScanBarcode	Prompt the user to scan a barcode label on the DUT. [***Deprecated***]
ScanLabel	Prompt the user to scan a label on the DUT. The results are stored in the specified variable.
SetSerialNumber	Store the DUT's unique serial number and use that serial number for all database logging.. [***Required***]
SystemCommand	Execute a system command.

5.13.2.1 Connect

Establish a connection to a DUT.

Attributes

Name	The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.
Validate	Verify that communication is setup with the right type of DUT. Permissible values are "Yes" or "No". [Not implemented in this release.]
Timeout	The number of seconds to wait for a response before aborting the action.

Example

```
<Connect Name="Establish Connection" Validate="Yes" Timeout="30" />
```

5.13.2.2 Disconnect

Disconnect from a DUT.

Attributes

Name	The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

Example

```
<Disconnect Name="Disconnect from DUT" />
```

5.13.2.3 DownloadImage

Download an image from the host to the DUT. Typical uses are to download a firmware image or application image to be programmed into flash memory. Any other type of binary image can also be downloaded and programmed into any programmable logic device such as an FPGA, CPLD, or other device. This action provides a very fast means of transferring and programming a new image during production runs or during interactive validation. The maximum length of the filename, including extension, is limited to 14 characters.

Attributes

- **Name** – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.
- **Timeout** – The number of seconds to wait for a response before aborting this action.

Additional Tags

- **<Path>** – The full path name of the file to download. The path is relative to the TFTP folder specified in kozioserver.properties, such as "myTests".

The actual filename, including extension, must be 14 characters or less.

<ServerIP> – The IP address of the server that will perform the TFTP transfer.

<Output> – A variable to store results.

Example

```
<DownloadImage Name="Simple Image Download" Timeout="30">
  <Path>customapp.bin</Path>
  <ServerIP>192.168.1.32</ServerIP>
  <Output>dummy-output</Output>
</DownloadImage>
```

5.13.2.4 DownloadScript

Download a kScript file to the DUT using a running TFTP server. This is the most efficient means of transferring an entire script file for later execution. The script file is sent in a single transfer to the DUT and interpreted. It is best for the script file to contain definitions of new commands for later execution. The maximum length of the filename, including extension, is limited to 14 characters.

Attributes

Name – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

Timeout – The number of seconds to wait for a response before aborting this action.

Additional Tags

<Path> – The full path name of the file to download. The path is relative to the TFTP folder specified in kozioserver.properties, such as "myTests". The actual filename, including extension, must be 14 characters or less.

<ServerIP> – The IP address of the server that will perform the TFTP transfer.

<Output> – A variable to store results.

Example

```
<DownloadScript Name="Simple kScript Command" Timeout="90">
  <Path>Report.ksc</Path>
  <ServerIP>192.168.1.32</ServerIP>
  <Output>dummy-output</Output>
</DownloadScript>
```

5.13.2.5 ExecuteCommand

Command kDiagnostics® to execute a test command on the DUT.

Attributes

Name – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

Timeout – The number of seconds to wait for a response before aborting this action.

Additional Tags

<Command> – The kDiagnostics® command to be executed. The command must exist in kDiagnostics® either from the default list of commands provided by Kozio or from a user defined command.

<FilePath> – The filename of the script file to download to the DUT. The actual filename, including extension, must be 14 characters or less.

Note: See Example 2 below, this shows a script file is sent to the DUT defining a new test suite name and then that test suite is executed using second ExecuteCommand action.

<Input> – The value of the variable is used as the command to execute on the DUT. This tag is optional.

Note: See the Example 3 below where the variable "kScriptCommand" is set using a Prompt action and then passed to the ExecuteCommand action.

<Output> – A variable to store results.

Example 1

```
<ExecuteCommand Name="kScript Command - test.sdram" Timeout="30">
    <Command>test.sdram.quick</Command>
    <Output>script-results</Output>
</ExecuteCommand>
```

Example 2

```
<ExecuteCommand Name="kScript Command - test.custom.final"
Timeout="30">
    <Command />
    <FilePath>Mfg_Test.ksc</FilePath>
    <Output>script-results</Output>
</ExecuteCommand>

<ExecuteCommand Name="kScript Command - test.custom.final"
Timeout="60">
    <Command>test.custom.final</Command>
    <Output>script-results</Output>
</ExecuteCommand>
```

Example 3

```
<Prompt Name="Prompt to Execute Test Command" OkCancel="Yes"
Timeout="30">
    <Title>Prompt to Execute Test Command</Title>
    <MessageText><![CDATA[<html><b>Enter a test command to execute on
the UUT.<br>
    <br>
      Type a kScript command into the prompt box below.<br>
      <br>
      This command will be sent to the UUT and executed.<br>
      <br>
      This demonstrates the use of passing variables from one Action
to another.</b></html.]]></MessageText>
      <Output>kScriptCommand</Output>
</Prompt>
```

```
<Connect Name="Connect To DUT" Validate="Yes" Timeout="30" />

<ExecuteCommand Name="Execute kScript Command" Timeout="15">
    <Command />
    <Input>kScriptCommand</Input>
    <Output>script-results</Output>
</ExecuteCommand>
```

5.13.2.6 GetDatabase

This test action retrieves a value from a database table. Use this action to retrieve values from the Common or DUT tables.

Attributes

Name – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

Additional Tags

<Table> – Read a value from this database table, with valid values of "common" or "dut".

<RowName> – The label used to identify a given row of the table. See the PutDatabase action for how to create or update rows.

<FieldNumber> – Read this user field number from the row. Valid values are 1 to 20.

<Output> – A variable to store results.

Example

```
<GetDatabase Name="GetDatabase Test Action">
    <Table>common</Table>
    <RowName>DebbyTest0022</RowName>
    <FieldNumber>3</FieldNumber>
    <Output>GetDatabaseOutput</Output>
</GetDatabase>
```

5.13.2.7 GetFile

This test action reads the contents of a disk file and stores that data in a variable.

Attributes

Name – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

FileDialog – This optional attribute specifies whether to use a file browser for selecting which file to be read. Valid options are "yes" or "no" and the default is "no", do not use a file browser.

Additional Tags

<Output> – A variable that stores what was read from the file.

<FilePath> – Optional field denotes the file path relative to TftpRoot value, which is defined in kozioserver.properties. If omitted, file will be placed in TftpRoot.

<FileName> – The name of the file to be read.

Example

```
<!-- read a file, browse for the filename -->
```

```
    <GetFile Name="Read Disk File" FileDialog="yes">
    <Output>GetFileContent</Output>
    <FileName>GetFileTest.xml</FileName>
</GetFile>

<!-- display the file contents -->
<Message Name="Data Display" OkCancel="yes">
    <Title>File Data</Title>
    <MessageText><![CDATA[<html><b>
      The data read was: </b><br>
      %GetFileContent%.<br></html.]]>
    </MessageText>
</Message>

<!-- read a file, with no browsing -->
<GetFile Name="Read Disk File" FileDialog="no">
    <Output>GetFileContent</Output>
    <FileName>example.txt</FileName>
</GetFile>

<!-- display the file contents -->
<Message Name="GetFile Data Display" OkCancel="yes">
    <Title>File Data</Title>
    <MessageText><![CDATA[<html><b>
      The data read was: </b><br>
      %GetFileContent%.<br>
      </html.]]></MessageText>
</Message>
```

5.13.2.8 GetValue

This test action retrieves a value from a variable by parsing the text. Use this action to retrieve values from barcode labels and strings stored in user fields in the database.

Attributes

Name – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

Additional Tags

<Title> – The text that will appear on the dialog's title bar.

<Input> – The variable value to parse.

<LineNumber> – The line of the text to parse. Line numbers start at 1 in the text.

<StartOffset> – The first character in the text to extract. The first character is offset 0.

<SubstringLength> – The number of characters in the text to extract. Parsing will stop once the end of line character is reached even if the number of characters read is less than the value specified here. This allows variable length values to be read with one test sequence file as long as each variable is on a separate line.

<Output> – A variable to store results.

Example

```
<GetValue Name="GetValue Action">
    <Title>Parse Scanned Label</Title>
    <Input>ScannedLabel</Input>
    <LineNumber>1</LineNumber>
    <StartOffset>3</StartOffset>
    < SubstringLength >127</SubstringLength>
    <Output>SerialNumber</Output>
</GetValue>
```

5.13.2.9 Message

Display a message to the operator.

Attributes

Name – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

OkCancel – The option to display "OK" and "Cancel" buttons. Permissible values are "Yes" or "No".

Additional Tags

<Title> – The text that will appear on the dialog's title bar.

<MessageText> – The text that will appear in the dialog box. Either a simple text or HTML formatted text can be used that is contained within a "<![CDATA[…]]>" tag construct.

Example

```
<Message Name="Welcome" OkCancel="Yes">
    <Title>Welcome!</Title>
    <MessageText><![CDATA[<html><font size="+1" face="Verdana">
      <br>
      <center><u>Operator</u></center><br>
    Plug the phone cable into voice port one (1).<br>
      <br>
      Click OK to proceed.<br>
      <br>
    </b></font></html]]>
    </MessageText>
</Message>
```

Figure 5.8. Message Action from the Example

5.13.2.10 OperatorComment

Prompt the operator to enter a comment. The comment is stored in the database as part of the test results record. This action must be placed after actions for executing commands, for example the ExecuteCommand action.

Attributes

- **Name** – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.
- **OkCancel** – The option to display "OK" and "Cancel" buttons. Permissible values are "Yes" or "No".
- **AskWhen** – Defines when the operator should be prompted to enter a comment into the test database. Permissible values are "OnError" or "Always".
- **Timeout** – The number of seconds to wait for a response before aborting this action.

Additional Tags

- **<Title>** – The text that will appear on the dialog's title bar.
- **<MessageText>** – The text that will appear in the dialog box. Either a simple text or HTML formatted text can be used that is contained within a "<![CDATA[…]]>" tag construct. This example uses simple text.

Example

```
<OperatorComment Name="Enter Comment" OkCancel="Yes" AskWhen="OnError"
                 Timeout="30">
  <Title>Enter Comment</Title>
  <MessageText>Enter a comment regarding the nature of the
failure!</MessageText>
</OperatorComment>
```

Below is what the above example looks like with some operator text filled in.

Fixture 1 - Enter Comment
Please enter a comment regarding the nature of the failure.
I entered an invalid DUT test command.
OK
Cancel

Figure 5.9. OperatorComment Action Example

5.13.2.11 Prompt

Prompt the operator for input.

Attributes

- **Name** – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.
- **OkCancel** – The option to display "OK" and "Cancel" buttons. Permissible values are "Yes" or "No".
- **Timeout** – The number of seconds to wait for a response before aborting this action.

Additional Tags

- **<Title>** – The text that will appear on the dialog's title bar.
- **<Image>** – An image to display.
- **<MessageText>** – The text that will appear in the dialog box. Either a simple text or HTML formatted text can be used that is contained within a "<![CDATA[…]]>" tag construct.
- **<Output>** – A variable to store results.

Example

```
<Prompt Name="Prompt with image display" OkCancel="Yes" Timeout="30">
    <Title>Prompt with image display</Title>
    <Image>example.jpg</Image>
    <MessageText><![CDATA[<html><b>This prompt displays a picture.<br>
    <br>
      There are several options with prompt actions:
      <ul>
      <li>Display <b>with</b> an image</li>
      <li>Display <b>without</b> an image</li>
      <li>Display and receive user input from a <b>barcode
reader</b></li>
      <li>Display and receive user input from a <b>text box</b></li>
      <li>Store data in a variable to pass to other actions</li>
      </ul>
      <p>
      Select OK to continue<br>
    </b></html.]]></MessageText>
    <Output>kScriptCommand</Output>
</Prompt>
```

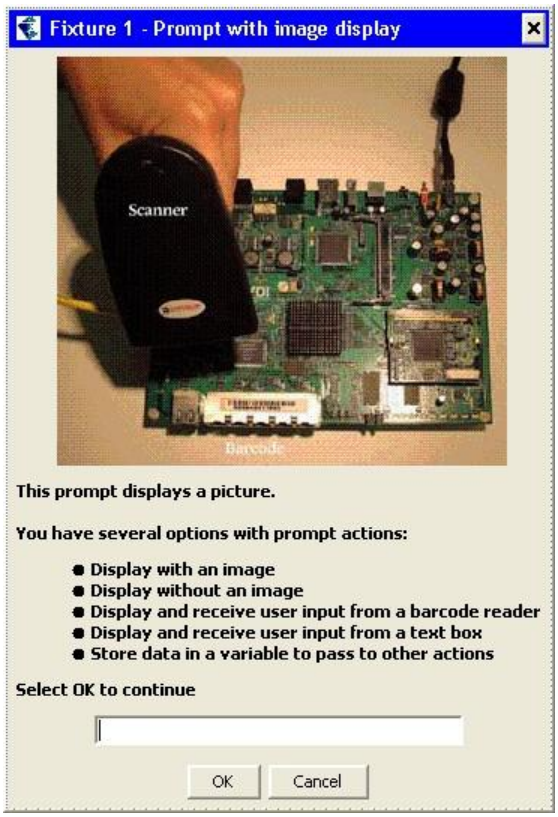

Fixture 1 - Prompt with image display
Scanner
This prompt displays a picture.
You have several options with prompt actions:
Display with an image
Display without an image
Display and receive user input from a barcode reader
Display and receive user input from a text box
Store data in a variable to pass to other actions
Select OK to continue
OK
Cancel

Figure 5.10. Prompt Action from previous Example

5.13.2.12 PutDatabase

This test action stores a value in a database table; either in the Common or DUT table may be specified.

Attributes

Name – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

Additional Tags

<Table> – Specify the database table, with valid values of "common" or "dut".

<RowName> – The label used to identify a given row of the table. If RowName does not exist, a new row is created. If the RowName exists, the row is updated. This field only applies to the **Common** table. For the **DUT** table, we use the row associated with the DUT currently being tested. (An error is generated if SetSerialNumber has not been executed to assign a serial number with a DUT.)

<FieldNumber> – Read this user field number from the row. Valid values are 1 to 20.

<Input> – A variable storing a value to be placed in the database.

Example

```
<PutDatabase Name="PutDatabase Test Action">
    <Table>common</Table>
    <RowName>DebbyTest002</RowName>
    <FieldNumber>3</FieldNumber>
    <Input>ScannedLabel</Input>
</PutDatabase>
```

5.13.2.13 PutFile

This test action saves the contents of a sequence variable to a disk file. This action is useful for exporting database contents or variable values.

Attributes

Name – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

Additional Tags

<Input> – A variable storing a value that will be written to the file.

<FilePath> – This optional field denotes the file path relative to the TftpRoot value (which is defined in kozioserver.properties). If omitted, the file will be placed in TftpRoot.

<FileName> – The name of the file to be created or overwritten.

<Input> – A variable storing a value to be placed in the database.

Example

```
<!-- Get the serial number for test documentation and device
programming -->
<ScanLabel Name="Input Test Script" OkCancel="yes">
    <Title>Scan Label</Title>
    <MessageText><![CDATA[<html><b>
      This test uses a 2D scanner to scan a 1D label. <br>Scan the
label
        when ready and click OK when done:</b></html.]]></MessageText>
    <LineCount>2</LineCount>
    <Output>ScannedText</Output>
</ScanLabel>

<PutFile Name="Save to Disk">
    <Input>ScannedText</Input>
    <FilePath>label</FilePath>
    <FileName>label_info.txt</FileName>
</PutFile>
```

5.13.2.14 ScanBarcode

Receive input from a barcode reader. It is recommended that you use ScanLabel in new sequence files as a replacement for ScanBarcode. Note that ScanBarcode requires SetSerialNumber.

Attributes

Name – Name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

OkCancel – The option to display "OK" and "Cancel" buttons. Permissible values are "Yes" or "No".

Timeout – Number of seconds to wait for a response before aborting this action.

Additional Tags

<Title> – The text that will appear on the dialog's title bar.

<Image> – An image to display.

<MessageText> – The text that will appear in the dialog box. Either a simple text or HTML formatted text can be used that is contained within a "<![CDATA[…]]>" tag construct.

<Output> – A variable to store results.

Example

```
<ScanBarcode Name="Scan Barcode to Start Test" OkCancel="Yes"
Timeout="30">
    <Title>Scan Barcode</Title>
    <Image>images\\example.jpg</Image>
    <MessageText><![CDATA[<html><font size="+1" face="Verdana">
      <br>Verify that all cables are connected and power on the
unit.<br>
      <br>
      <b>Scan the barcode to proceed.<br></b></html.]]>
    </MessageText>
    <Output>SerialNumber</Output>
</ScanBarcode>
```

5.13.2.15 ScanLabel

Receive input from a barcode reader and store the results in a variable. Use this action to retrieve barcode text to parse using the GetValue action or store in the database using the PutDatabase action.

Attributes

Name – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

OkCancel – The option to display "OK" and "Cancel" buttons. Permissible values are "yes" or "no". When "yes" is chosen, the operator is forced to select a button to continue the test sequence. When "no" is chosen, the dialog disappears after <LineCount> number of lines are read.

Additional Tags

<Title> – The text that will appear on the dialog's title bar.

<MessageText> – The text that will appear in the dialog box. Use simple text or HTML formatted text that is contained within a "<![CDATA[…]]>" tag construct.

<LineCount> – The number of text lines to read from the barcode label. When the line count is reached, no more input is stored in the variable, and the dialog is closed. This variable allows the test sequence to continue automatically once a barcode is scanned, instead of the operator selecting the OK button.

<Output> – A variable to store results.

Example

```
<ScanLabel Name="Enter Serial Number" OkCancel="no">
```

```
    <Title>Scan Label</Title>
    <MessageText><![CDATA[<html><b>
        This test uses the 2D scanner to scan a 1D ADI label. <br>
        Scan the label when ready and click OK when done:
        </b></html.]]>
    </MessageText>
    <LineCount>2</LineCount>
    <Output>ScannedLabel</Output>
</ScanLabel>
```

5.13.2.16 SetSerialNumber

This action stores a serial number used for all DUT related database queries. This action is required, even is a default serial number is used for all DUTs. If each DUT has a unique serial number, that value can be used for report generation.

Attributes

Name – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

Additional Tags

<Title> – The text that will appear on the dialog's title bar.

<Input> – A variable containing the serial number used to identify uniquely a DUT.

Example

```
<SetSerialNumber Name="SetSerialNumber Action">
    <Title>Set Serial Number</Title>
    <Input>SerialNumber</Input>
</SetSerialNumber>
```

5.13.2.17 SystemCommand

Execute a command on the host computer. This command can be used to perform any number of tasks using simple programs. Possible tasks could include executing database queries, using the Command Line Interface to execute DUT commands as shown below, or using a program to convert a barcode value into other data.

Attributes

Name – The name of this action as it will appear after "Current Action:" on the SequenceRunner™ user interface screen.

Additional Tags

<Command> – The system command to be executed.

<Output> – A variable to store results.

Example

```
<SystemCommand Name="Example of External Command Execution">
    <Command>cli -s Report.ksc</Command>
```

```
        <Output>system-cmd-results</Output>
    </SystemCommand>
```

5.13.3 Passing Values between Test Sequence Actions

Pass values from one action to the next using the <Output> and <Input> variable tags. To pass a value, declare a variable name using the <Output> tag in one action, for example:

```
<Output>kScriptCommand</Output>
```

The output from that action tag is stored in the variable. Access that value using the <Input> tag in a subsequent action tag. Such as:

```
<Input>kScriptCommand</Input>
```

The name of the variable must match exactly and is case sensitive. See ExecuteCommand for an example.

6
SequenceReporter™

SequenceReporter™ allows the generation, saving, and printing of test reports for publishing from any LAN connection. Given the right networking set up, checks on contract manufacturers' test results over a WAN connection can also be made. The **SequenceReporter™** user interface allows an engineer or manager instant access to circuit board test results and generates queries to see full circuit board test results. Report data includes board type, serial #, test date, operator, station, operator notes, and a complete test summary. Launch SequenceReporter™ using the desktop shortcut.

SequenceReporter™ displays a startup banner followed by the main operator interface.

Figure 6.1. SequenceReporter™ – Startup Banner

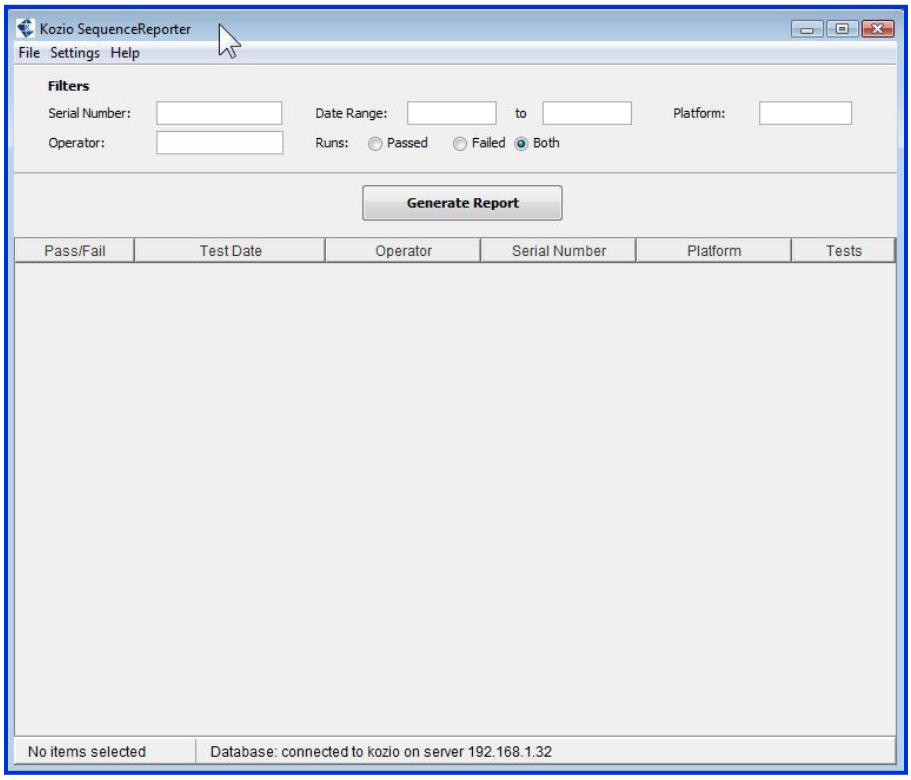


Figure 6.2. SequenceReporter™ – Initial Screen

In old version of screen above, Database connected message was green.

When SequenceReporter™ runs, it uses saved values from a prior run and automatically attempts to connect to a database. The database connection status is show on the status line at the bottom of the window. In this screen shot, we see we have a successful connection.

6.1 Switching Database Settings

Select the menu "Settings → Database" to update database settings.

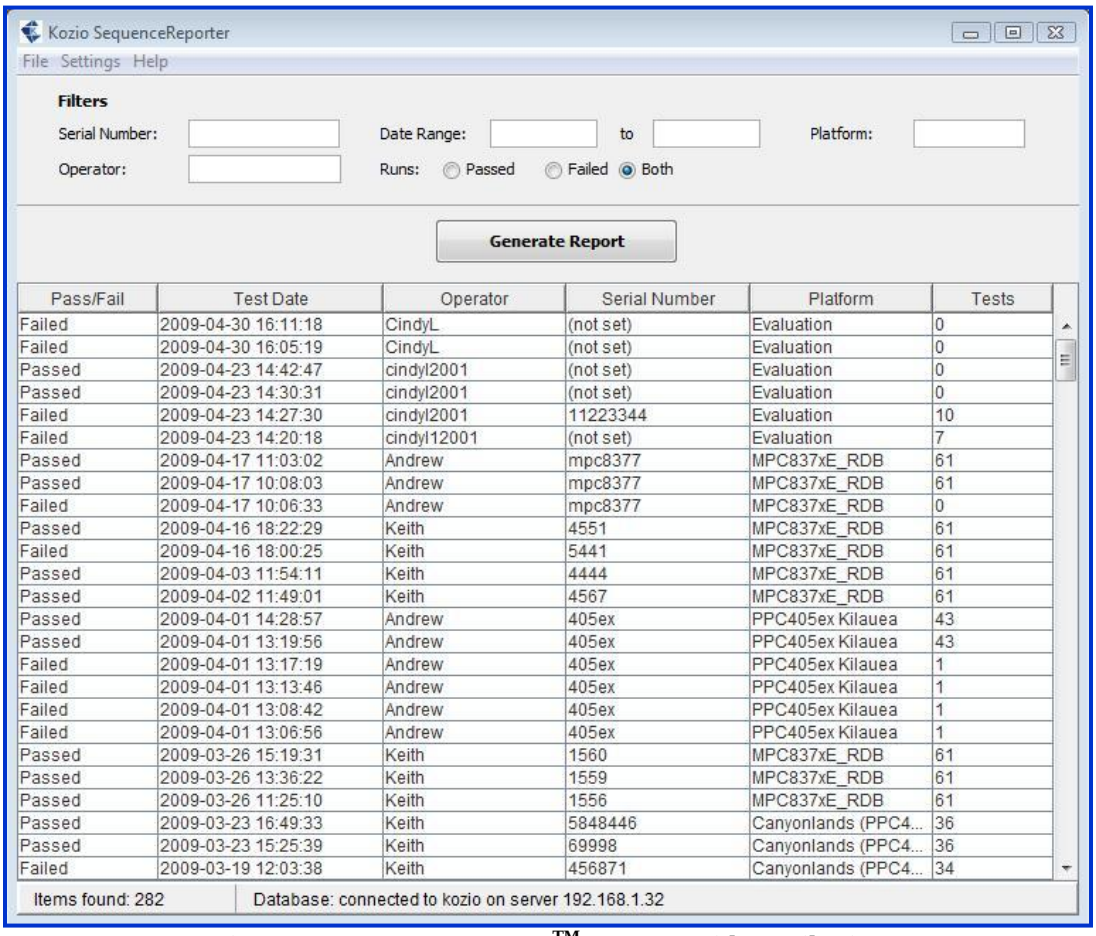
Figure 6.3. SequenceReporter™ – Database Settings Dialog

Fill in "***Database Server***" with the IP address of the machine running the PostgreSQL database. The "***Database Name***", "***User name***", and "***Password***" fields must be filled in with values entered during the database installation process.

Select the "***Save***" button to update the stored settings. Select "***Connect***" to attempt a connection to the database using the settings that have been entered.

6.2 Generate Report

Once connected to a database, select the "Generate Report" button to query the database using the filter values that have been entered. If all filters are blank, all records from the database are retrieved and displayed.



Pass/Fail	Test Date	Operator	Serial Number	Platform	Tests
Failed	2009-04-30 16:11:18	CindyL	(not set)	Evaluation	0
Failed	2009-04-30 16:05:19	CindyL	(not set)	Evaluation	0
Passed	2009-04-23 14:42:47	cindyl2001	(not set)	Evaluation	0
Passed	2009-04-23 14:30:31	cindyl2001	(not set)	Evaluation	0
Failed	2009-04-23 14:27:30	cindyl2001	11223344	Evaluation	10
Failed	2009-04-23 14:20:18	cindyl12001	(not set)	Evaluation	7
Passed	2009-04-17 11:03:02	Andrew	mpc8377	MPC837xE_RDB	61
Passed	2009-04-17 10:08:03	Andrew	mpc8377	MPC837xE_RDB	61
Failed	2009-04-17 10:06:33	Andrew	mpc8377	MPC837xE_RDB	0
Passed	2009-04-16 18:22:29	Keith	4551	MPC837xE_RDB	61
Failed	2009-04-16 18:00:25	Keith	5441	MPC837xE_RDB	61
Passed	2009-04-03 11:54:11	Keith	4444	MPC837xE_RDB	61
Passed	2009-04-02 11:49:01	Keith	4567	MPC837xE_RDB	61
Passed	2009-04-01 14:28:57	Andrew	405ex	PPC405ex Kilauea	43
Passed	2009-04-01 13:19:56	Andrew	405ex	PPC405ex Kilauea	43
Failed	2009-04-01 13:17:19	Andrew	405ex	PPC405ex Kilauea	1
Failed	2009-04-01 13:13:46	Andrew	405ex	PPC405ex Kilauea	1
Failed	2009-04-01 13:08:42	Andrew	405ex	PPC405ex Kilauea	1
Failed	2009-04-01 13:06:56	Andrew	405ex	PPC405ex Kilauea	1
Passed	2009-03-26 15:19:31	Keith	1560	MPC837xE_RDB	61
Passed	2009-03-26 13:36:22	Keith	1559	MPC837xE_RDB	61
Passed	2009-03-26 11:25:10	Keith	1556	MPC837xE_RDB	61
Passed	2009-03-23 16:49:33	Keith	5848446	Canyonlands (PPC4...	36
Passed	2009-03-23 15:25:39	Keith	69998	Canyonlands (PPC4...	36
Failed	2009-03-19 12:03:38	Keith	456871	Canyonlands (PPC4...	34

Figure 6.4. SequenceReporter™ – Report with no Filtering

Use **Serial Number**, **Date Range**, **Platform**, **Operator**, and **Runs** filters to reduce the test results displayed.

Filter	Description
Serial Number	A text string representing a serial number.
Date Range	Formats that are accepted: YYYY-MM-DD HH:MM MM-DD-YY HH:MM MM-DD-YYYY HH:MM Representing year, month, day, hour minute, and hours and minutes are optional. Dashes (-) or slashes (/) can be used between the month, day, and year.
Platform	Text string representing platform name - board's name or project's name.
Operator	A text string for the operator's name.
Runs	Select to view only **Passed** runs, only **Failed** runs, or **Both**.

Select "Generate Report" or press the Enter key to update the query results after the filter settings have been modified. Here is an example with filtering.

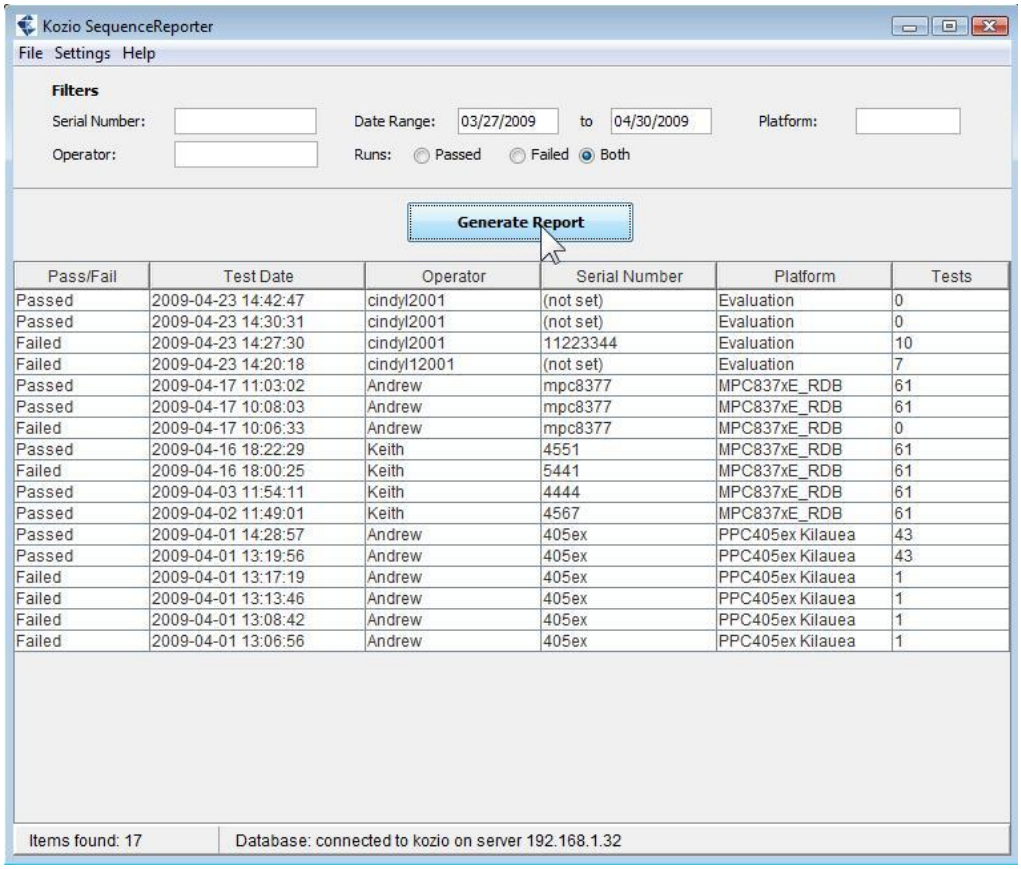

Kozio SequenceReporter
File Settings Help
Filters
Serial Number:
Date Range: 03/27/2009 to 04/30/2009
Platform:
Operator:
Runs: Passed Failed Both
Generate Report
Pass/Fail | Test Date | Operator | Serial Number | Platform | Tests
Passed | 2009-04-23 14:42:47 | cindyl2001 | (not set) | Evaluation | 0
Passed | 2009-04-23 14:30:31 | cindyl2001 | (not set) | Evaluation | 0
Failed | 2009-04-23 14:27:30 | cindyl2001 | 11223344 | Evaluation | 10
Failed | 2009-04-23 14:20:18 | cindyl12001 | (not set) | Evaluation | 7
Passed | 2009-04-17 11:03:02 | Andrew | mpc8377 | MPC837xE_RDB | 61
Passed | 2009-04-17 10:08:03 | Andrew | mpc8377 | MPC837xE_RDB | 61
Failed | 2009-04-17 10:06:33 | Andrew | mpc8377 | MPC837xE_RDB | 0
Passed | 2009-04-16 18:22:29 | Keith | 4551 | MPC837xE_RDB | 61
Failed | 2009-04-16 18:00:25 | Keith | 5441 | MPC837xE_RDB | 61
Passed | 2009-04-03 11:54:11 | Keith | 4444 | MPC837xE_RDB | 61
Passed | 2009-04-02 11:49:01 | Keith | 4567 | MPC837xE_RDB | 61
Passed | 2009-04-01 14:28:57 | Andrew | 405ex | PPC405ex Kilauea | 43
Passed | 2009-04-01 13:19:56 | Andrew | 405ex | PPC405ex Kilauea | 43
Failed | 2009-04-01 13:17:19 | Andrew | 405ex | PPC405ex Kilauea | 1
Failed | 2009-04-01 13:13:46 | Andrew | 405ex | PPC405ex Kilauea | 1
Failed | 2009-04-01 13:08:42 | Andrew | 405ex | PPC405ex Kilauea | 1
Failed | 2009-04-01 13:06:56 | Andrew | 405ex | PPC405ex Kilauea | 1
Items found: 17
Database: connected to kozio on server 192.168.1.32

Figure 6.5. SequenceReporter™ – Report with Filtering

6.3 Print Query Results

Use the File->Print Report menu option to print the query results. This saves the display on the main screen.

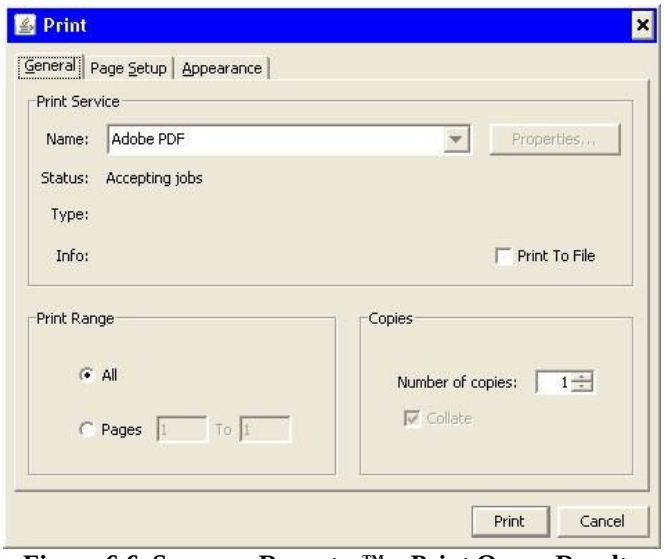


Figure 6.6. SequenceReporter™ – Print Query Results

6.4 Save Report

Additional options are available when an item displayed in the results pane is selected. Select a line item and use the right-mouse button to display a pop-up menu. Access to information stored in the database is now available. The stored Test Summary, Detailed Log, or Operator Comments can be viewed.

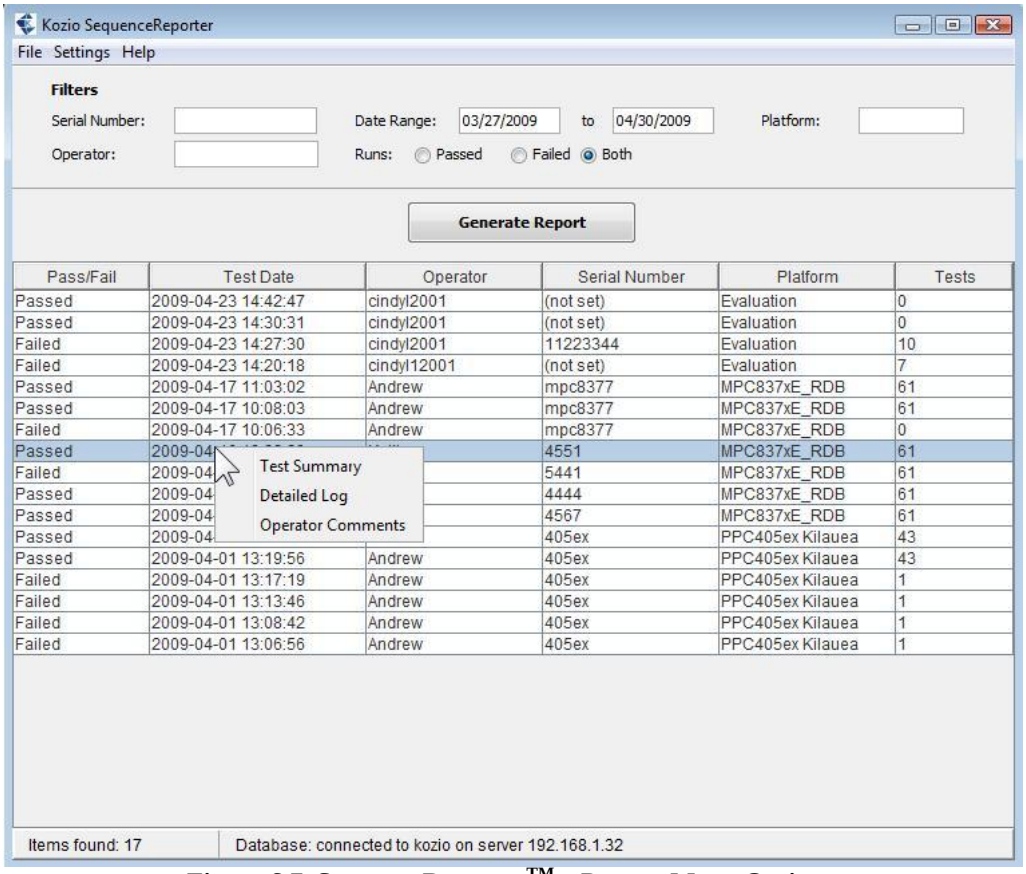

Kozio SequenceReporter
File Settings Help
Filters
Serial Number:
Date Range: 03/27/2009 to 04/30/2009
Platform:
Operator:
Runs: Passed Failed Both
Generate Report
Pass/Fail | Test Date | Operator | Serial Number | Platform | Tests
Passed | 2009-04-23 14:42:47 | cindyl2001 | (not set) | Evaluation | 0
Passed | 2009-04-23 14:30:31 | cindyl2001 | (not set) | Evaluation | 0
Failed | 2009-04-23 14:27:30 | cindyl2001 | 11223344 | Evaluation | 10
Failed | 2009-04-23 14:20:18 | cindyl12001 | (not set) | Evaluation | 7
Passed | 2009-04-17 11:03:02 | Andrew | mpc8377 | MPC837xE_RDB | 61
Passed | 2009-04-17 10:08:03 | Andrew | mpc8377 | MPC837xE_RDB | 61
Failed | 2009-04-17 10:06:33 | Andrew | mpc8377 | MPC837xE_RDB | 0
Passed | 2009-04 | | 4551 | MPC837xE_RDB | 61
Failed | 2009-04 | | 5441 | MPC837xE_RDB | 61
Passed | 2009-04 | | 4444 | MPC837xE_RDB | 61
Passed | 2009-04 | | 4567 | MPC837xE_RDB | 61
Passed | 2009-04 | | 405ex | PPC405ex Kilauea | 43
Passed | 2009-04-01 13:19:56 | Andrew | 405ex | PPC405ex Kilauea | 43
Failed | 2009-04-01 13:17:19 | Andrew | 405ex | PPC405ex Kilauea | 1
Failed | 2009-04-01 13:13:46 | Andrew | 405ex | PPC405ex Kilauea | 1
Failed | 2009-04-01 13:08:42 | Andrew | 405ex | PPC405ex Kilauea | 1
Failed | 2009-04-01 13:06:56 | Andrew | 405ex | PPC405ex Kilauea | 1
Test Summary
Detailed Log
Operator Comments
Items found: 17
Database: connected to kozio on server 192.168.1.32

Figure 6.7. SequenceReporter™ – Pop-up Menu Options

From the Test Summary dialog, the test report can be viewed and saves to a file. From the Detailed Log dialog, the detailed log can be viewed and saved it to a file. The Operator Comments dialog displays any notes entered by the operator for that test run. Here are samples from the three choices.

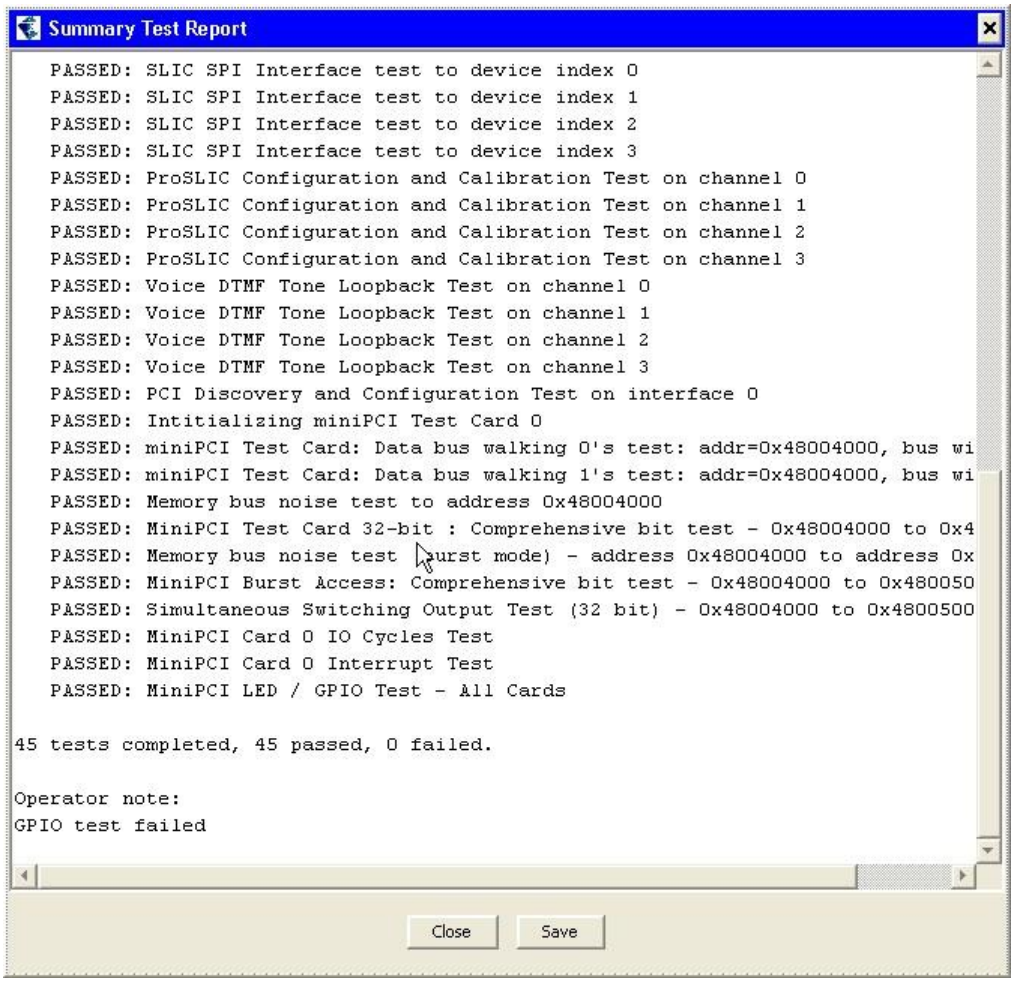

Summary Test Report
PASSED: SLIC SPI Interface test to device index 0
PASSED: SLIC SPI Interface test to device index 1
PASSED: SLIC SPI Interface test to device index 2
PASSED: SLIC SPI Interface test to device index 3
PASSED: ProSLIC Configuration and Calibration Test on channel 0
PASSED: ProSLIC Configuration and Calibration Test on channel 1
PASSED: ProSLIC Configuration and Calibration Test on channel 2
PASSED: ProSLIC Configuration and Calibration Test on channel 3
PASSED: Voice DTMF Tone Loopback Test on channel 0
PASSED: Voice DTMF Tone Loopback Test on channel 1
PASSED: Voice DTMF Tone Loopback Test on channel 2
PASSED: Voice DTMF Tone Loopback Test on channel 3
PASSED: PCI Discovery and Configuration Test on interface 0
PASSED: Intitializing miniPCI Test Card 0
PASSED: miniPCI Test Card: Data bus walking 0's test: addr=0x48004000, bus wi
PASSED: miniPCI Test Card: Data bus walking 1's test: addr=0x48004000, bus wi
PASSED: Memory bus noise test to address 0x48004000
PASSED: MiniPCI Test Card 32-bit : Comprehensive bit test - 0x48004000 to 0x4
PASSED: Memory bus noise test (burst mode) - address 0x48004000 to address 0x
PASSED: MiniPCI Burst Access: Comprehensive bit test - 0x48004000 to 0x480050
PASSED: Simultaneous Switching Output Test (32 bit) - 0x48004000 to 0x4800500
PASSED: MiniPCI Card 0 IO Cycles Test
PASSED: MiniPCI Card 0 Interrupt Test
PASSED: MiniPCI LED / GPIO Test - All Cards
45 tests completed, 45 passed, 0 failed.
Operator note:
GPIO test failed
Close
Save

Figure 6.8. SequenceReporter™ – Summary Test Report Dialog

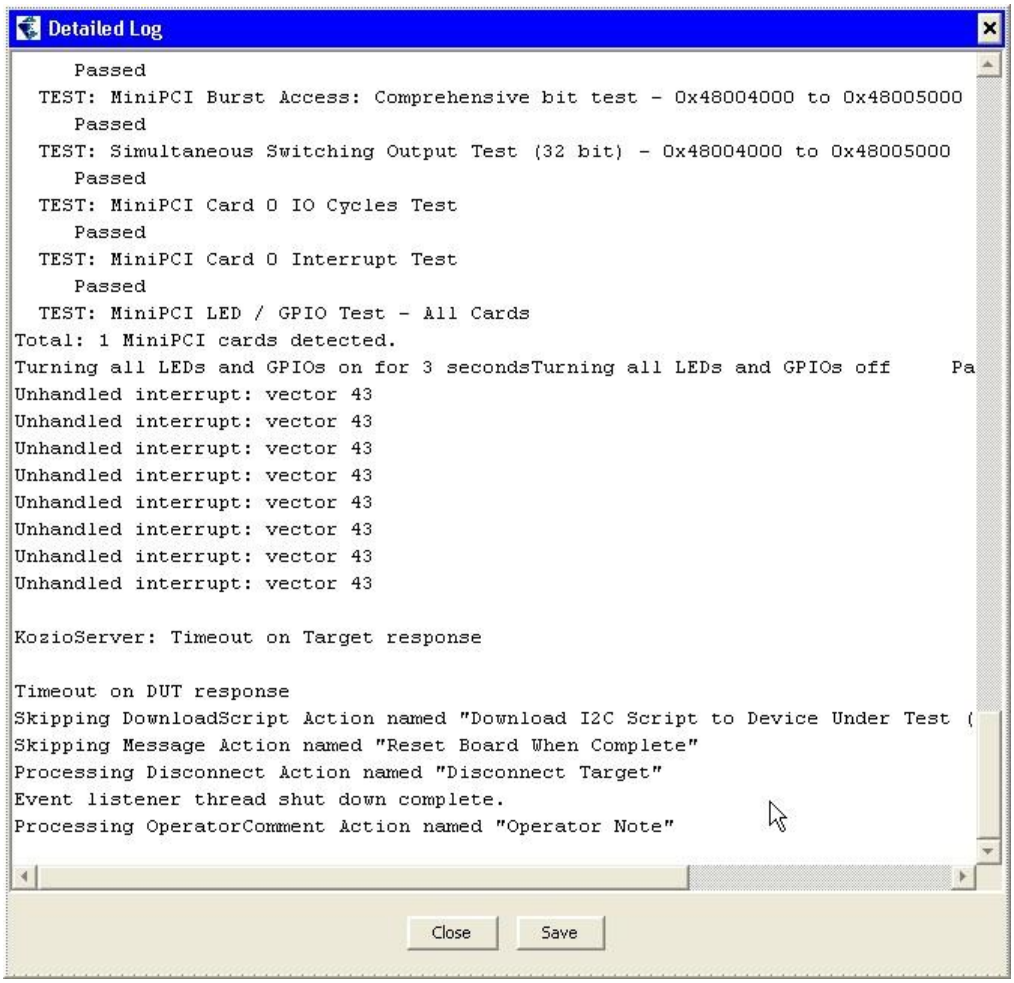

Detailed Log
Passed
TEST: MiniPCI Burst Access: Comprehensive bit test - 0x48004000 to 0x48005000
Passed
TEST: Simultaneous Switching Output Test (32 bit) - 0x48004000 to 0x48005000
Passed
TEST: MiniPCI Card 0 IO Cycles Test
Passed
TEST: MiniPCI Card 0 Interrupt Test
Passed
TEST: MiniPCI LED / GPIO Test - All Cards
Total: 1 MiniPCI cards detected.
Turning all LEDs and GPIOs on for 3 secondsTurning all LEDs and GPIOs off Pa
Unhandled interrupt: vector 43
Unhandled interrupt: vector 43
Unhandled interrupt: vector 43
Unhandled interrupt: vector 43
Unhandled interrupt: vector 43
Unhandled interrupt: vector 43
Unhandled interrupt: vector 43
Unhandled interrupt: vector 43
KozioServer: Timeout on Target response
Timeout on DUT response
Skipping DownloadScript Action named "Download I2C Script to Device Under Test (
Skipping Message Action named "Reset Board When Complete"
Processing Disconnect Action named "Disconnect Target"
Event listener thread shut down complete.
Processing OperatorComment Action named "Operator Note"
Close
Save

Figure 6.9. SequenceReporter™ – Detailed Log Dialog

Figure 6.10. SequenceReporter™ – Operator Comments Dialog

7
Multi-device Testing

Multi-device testing is an advanced feature of the kDiagnostics® Host Tools.

Multi-device configuration allows software tools running on a host computer to communicate with multiple devices under test. Each Host Tool (ValidationAssistant™, SequenceRunner™, and SequenceReporter™) provides the capability to select a Virtual Fixture (VF) and send test commands to the device connected to that fixture. See the specific sections for each Host Tool for details.

7.1 Multi-device Testing Equipment Virtual Fixture Set Up Example

The test set up described in Figure 1 is a configuration for testing four devices simultaneously. Complete configuration information is provided for the 16-port Switch and Virtual Fixtures.

1. Connect the 16-port switch to the test computer's NIC.
2. Connect VF-1's WAN port to the 16-port switch, repeat for VF-2, VF-3, and VF-4.
3. Connect VF-1's LAN port to DUT 1, repeat for VF-2, VF-3, and VF-4.
4. Enable power for the switch and each Virtual Fixture.
5. Apply power to each DUT.

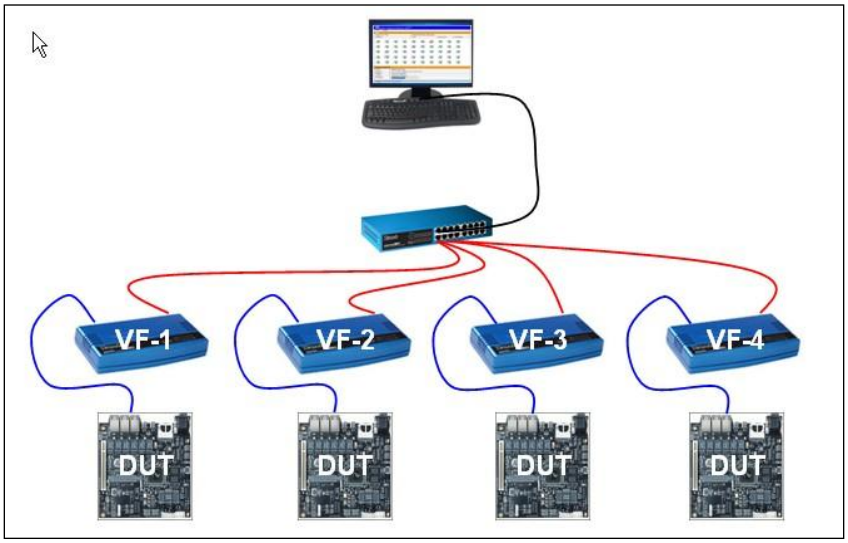


Figure 7.1 Four-DUT Virtual Fixture Configuration

7.2 Software Installation

The device under test must be running kDiagnostics® 2008.2 or later. kDiagnostics® must be running and the remote interface must be initialized. TheVirtual Fixtures feature eliminates the requirement to alter each DUT's default MAC Address and IP Address.

7.3 Properties File

The `kozioserver.properties` file contains configuration data linking a virtual fixture to an IP Address. Verify that this information matches the network configuration. The file is located in the root installation folder. The default location is: "C:\Program Files\Kozio".

The lines below are an example of the fixture to IP address declarations. If this file is altered, Kozio Server must be restarted.

```
NumberOfFixtures  = 4
Fixture1          = 192.168.1.200
Fixture2          = 192.168.1.201
Fixture3          = 192.168.1.202
Fixture4          = 192.168.1.203
```

7.4 SequenceRunner™

SequenceRunner™ provides a single instance user interface suitable for addressing all DUTs. SequenceRunner™ executes a selected test sequence. Sample test sequences are provided in the "examples" folder of the root installation directory.

SequenceRunner™ provides the ability to select a fixture and displays a status line at the bottom of the dialog box.

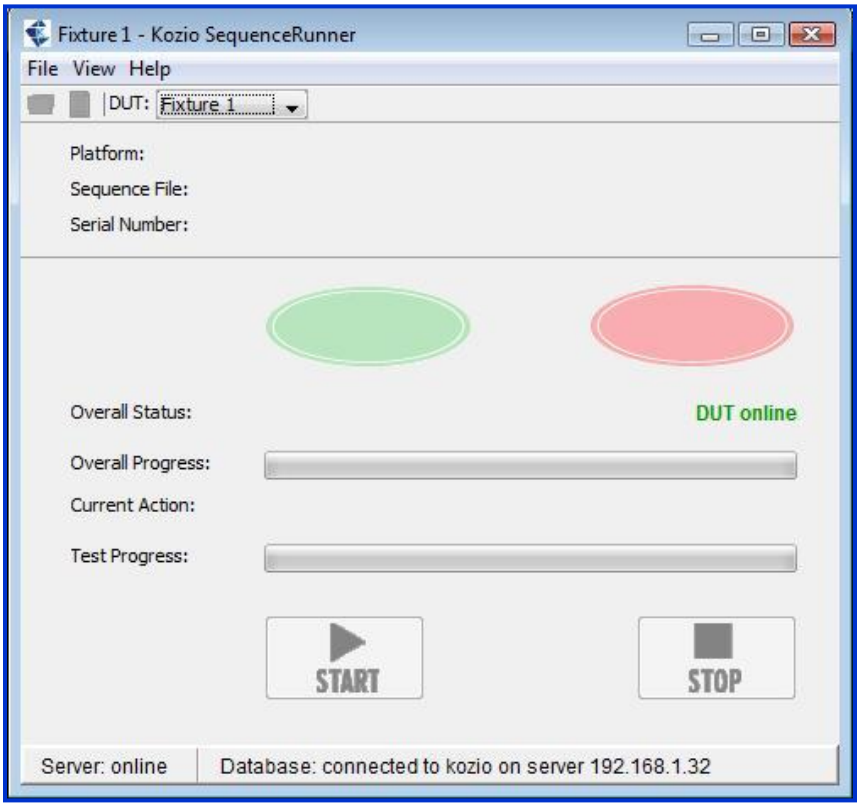


Figure 7.2 SequenceRunner™ with DUT Online

8 kHeader

Use the utility kHeader to wrap a binary file with header information before sending it to kDiagnostics® on the DUT. This feature ensures that a corrupted file is not programmed into flash memory.

The header of a wrapped file contains a CRC value for the binary image. kDiagnostics® will strip the header from the downloaded file and recalculate the CRC value for the file received. An error message is displayed if the file was corrupted during transmission.

Use kHeader to wrap all files before transmitting them to the DUT. Usage is displayed below.

```
Kozio kHeader v01
Usage: kHeader.exe -i INPUT-FILE -o OUTPUT-FILE -a DEST-ADDR -d DEST-TYPE
Creates a binary file with a header for use with the Kozio TFTP program.
    -i <file-name>           Use the specified file name as input
    -a <address>             Address to store the image in the target
    -d <dest-type>           Defines the type of memory in which the image
                             will be stored: 0=Flash, 1=EEPROM, 2=RAM, 3=NAND
    -o <file-name>           Use the specified file name for output
```

Figure 8.1. kHeader – Usage

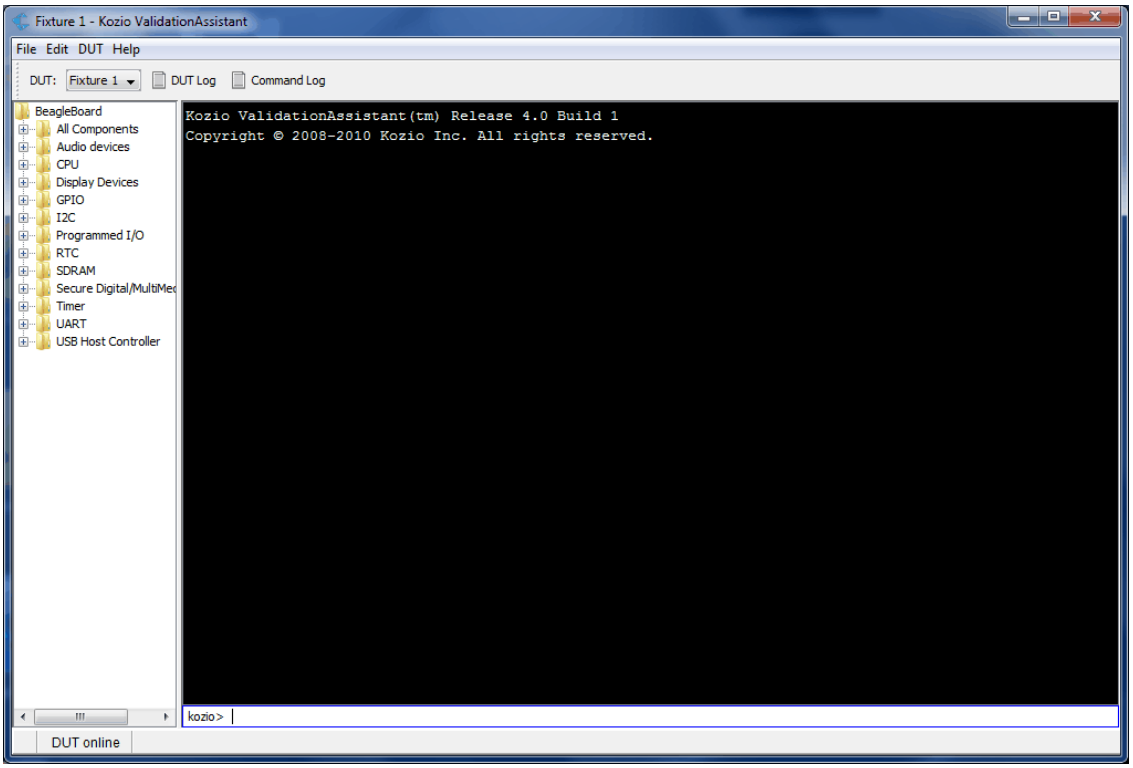


Figure 8-2Figure 8-3 ValidationAssistantTM with DUT Online

Enter the "**version**" command to see if the connection to the DUT is working properly. The command is then sent to the DUT. kDiagnostics® running on the DUT will respond with version and licensing information.

The following example depicts a DUT offline error message.

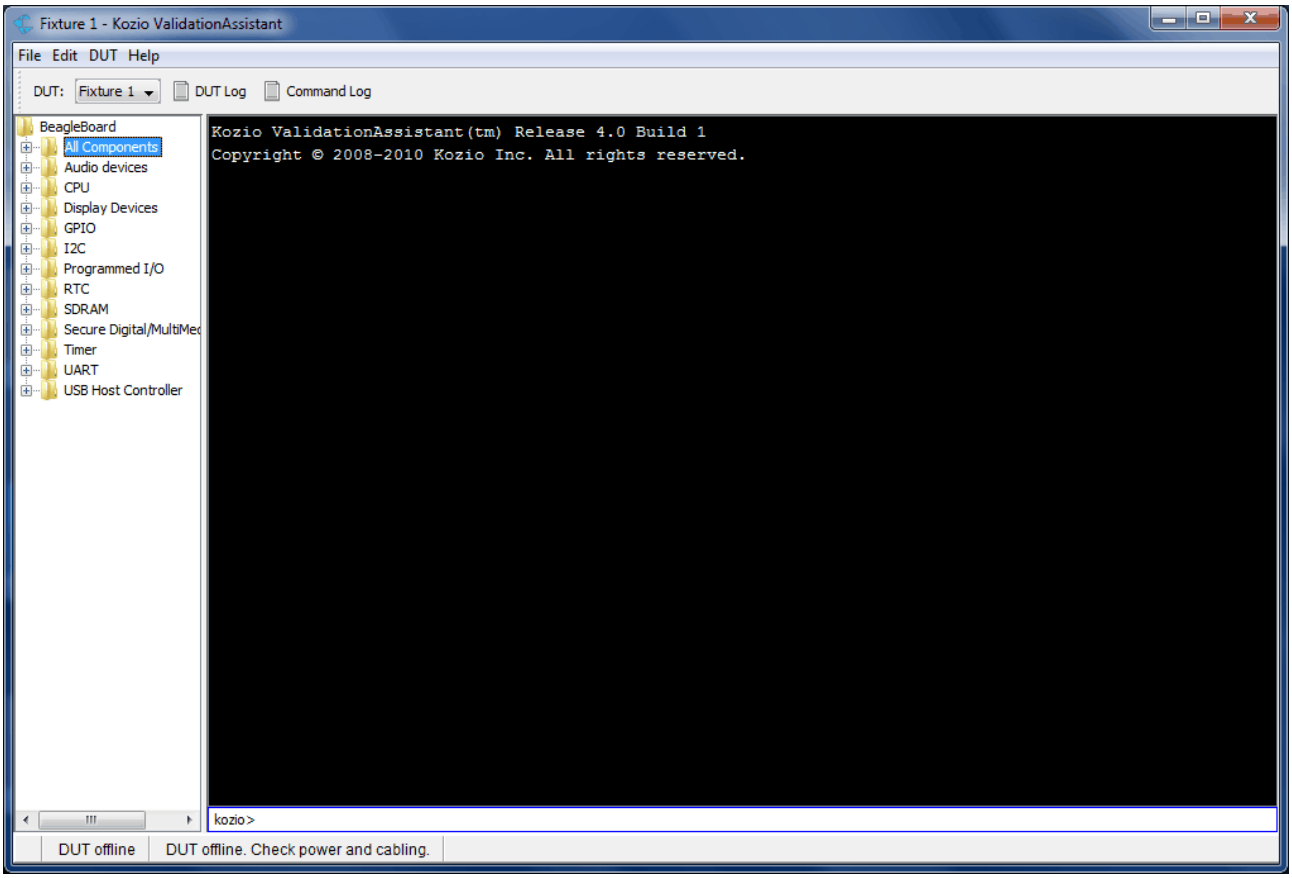


Figure 8-4Figure 8-5 ValidationAssistantTM 2.0 with DUT Offline

9

Troubleshooting

9.1 Timeout on command acknowledgement from DUT

When running ValidationAssistant™ and the error message "**KozioServer: Timeout on command acknowledgement from Target**" is displayed, this is indicative of a communications problem between KozioServer™ and DUT.

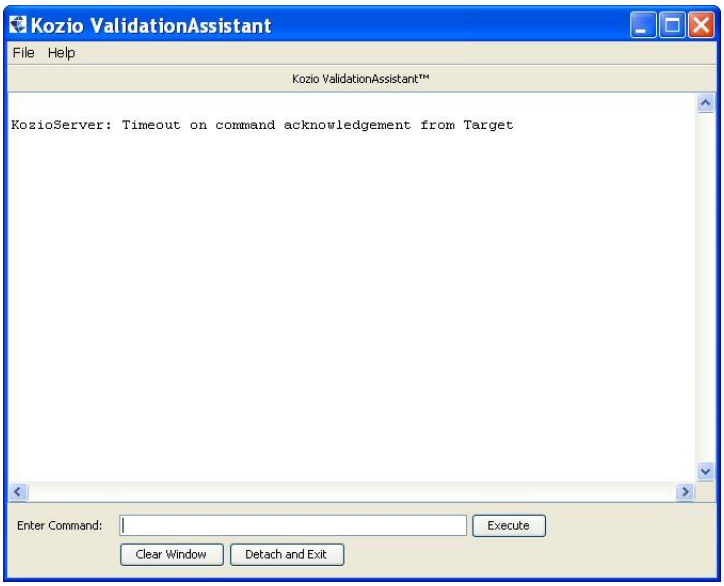


Figure 9-1. Error – KozioServerTM: Timeout on command acknowledgment from DUT

Here are a few troubleshooting tips:

- Make sure the DUT is powered on.
- From a Microsoft Command Prompt, ping the DUT to test the connection.
- Cycle power on the DUT in case kDiagnostics® is no longer communicating.
- Make sure that kDiagnostics® did not display a message on TeraTerm such as "**Warning: link is not up for PHY31**" This indicates that the wrong port is being used or the network connection is down.
- The DUT will display over the serial connection the message "**Target/Server handshake successful**" when the DUT is connected.
- Verify DUT and server IP address properties. kDiagnostics® will display remote communication properties on TeraTerm. KozioServer™ stores IP properties in the kozioserver.properties file.
- Make sure KozioServer™ is running. Under Windows, use the Start/Control Panel/Administrative Tools/Services to check KozioServer's status. Use Stop and then Start to restart KozioServer™. This is required any time kozioserver.properties is modified.
- Check the Windows Firewall settings to make sure that ports 4333, 4334 and 1099 are not blocked. These are the port numbers listed in the properties file. It may be necessary to disable the software firewall for testing.

9.2 Windows Security Alert

If a Windows Security Alert is displayed when running the RMI Server and the “Name” of the program is “Java(TM) Platform SE binary”, select the “Unblock” button.

Windows Security Alert
To help protect your computer, Windows Firewall has blocked some features of this program.
Do you want to keep blocking this program?
Name: Java(TM) Platform SE binary
Publisher: Sun Microsystems, Inc.
Keep Blocking
Unblock
Ask Me Later
Windows Firewall has blocked this program from accepting connections from the Internet or a network. If you recognize the program or trust the publisher, you can unblock it. When should I unblock a program?

Figure 9-2. Error – Windows Security Alert

To prevent this Alert from happening, follow the steps described in Section 2.3 to add the port to the Windows Firewall.

9.3 Windows Firewall Issues

Windows XP Firewall blocks by default the UDP and TCP communications ports used by the Kozio Software applications. If error messages about running any client application are encountered, follow the steps in Section 2.3 to update the Windows Firewall settings.

When the required ports are blocked, messages from the DUT are received be the host but blocked by the firewall. The client applications do not receive the expected messages. If the Kozio Host Tools do not receive messages, double check the Windows Firewall settings.

9.4 License Errors

A license to use the Host Tools is required. If a license file is missing, the following message appears.

Figure 9-3. Error – License file missing

Contact the Kozio Sales Department to purchase a license or contact Technical Support if a license has been purchased, but there are technical issues. Go to www.kozio.com/license/ to request a new license file.

The license file cannot be modified. If the file is altered, the following error message appears.

Figure 9-4. Error – License file invalid

To correct an invalid license issue replace the corrupted license file with the original license file. Contact Kozio Sales or Technical Support if the problem cannot be corrected.

10
Best Practices, Acronyms & Glossary

10.1 Best Practices

Here is a collection of items that should help overall efficiency using the Kozio Host Tools.

Manufacturing Suites When creating a manufacturing test suite, create a kScript file that contains the entire test configuration and test suites that are needed to validate each unit under test. Store this script in a file with a *.ksc extension. Create a test sequence file (*.xml) which will connect to the DUT. Download the kScript file, and execute the newly created test suite.

Using this method allows the tests executed on the DUT to be changed without changing the test sequence file.

Use one file or multiple file to store various test scripts. Create one test script file for validating the UUT and another test script file for downloading images and updating memories (Flash, EEPROM, and CPLD). Using multiple files allows separate actions to be taken and enhances reuse of the test script efforts for other projects.

10.2 Acronyms

ASCII	"American Standard Code for Information Interchange (ASCII) is a character encoding based on the English alphabet. ASCII codes represent text in computers, communications equipment, and other devices that work with text."
CLI	Command Line Interface
DUT	Device Under Test
DTD	Document Type Definition (DTD) is one of the XML schema languages used to detail the syntactical requirements of a given XML file type.
KSC	The file extension used to denote Kozio kScript files.
RMI	The Java Remote Method Invocation API, or Java RMI, is a Java application programming interface for performing remote procedure calls.
SLIP	The Serial Line Internet Protocol (SLIP) is an encapsulation of the Internet Protocol designed to work over serial ports. It is documented in RFC 1055. SLIP is often used with microprocessors, instead of using the Point-to-Point Protocol (PPP), due to its very small overhead.
XML	The Extensible Markup Language (XML) is a general-purpose markup language, much like HTML. It allows its users to define their own tags and enables the structured sharing of data.

10.3 Glossary

Connection	This describes the physical cabling between the host computer and DUT. The connection types may be a serial cable, Ethernet cable, or other type of supported

	connection.
Cygwin	Linux-like environment for Windows making it possible to port software running on POSIX systems (such as Linux, BSD, and Unix systems) to Windows. See www.**cygwin**.com for additional information.
Ethernet	"Ethernet is a family of frame-based computer networking technologies for local area networks (LANs). The name comes from the physical concept of the ether. It defines a number of wiring and signaling standards for the physical layer, through means of network access at the Media Access Control (MAC)/Data Link Layer, and a common addressing format. Ethernet is standardized as IEEE 802.3."
Host	A host computer is any computer used to run the software tools. This is a typical computer running Windows or Linux operating systems. The host computer may be connected to a network or may be connected directly to one or more DUTs.
IP Address	Acronym for Internet Protocol address. IP is the lower-level protocol within TCP/IP that is used to send data between computers over a network of computers. An IP address is an assigned number used to identify a computer that is connected to a network through TCP/IP. An IP address consists of four numbers (each of which can be no greater than 255) separated by periods, such as 192.168.1.1.
kScript	The scripting language interpreted by kDiagnostics®. A separate document that describes this scripting language is available.
Net Mask	Typically, a subnet represents all the machines at one geographic location, in one building, or on the same local area network (LAN). Having an organization's network divided into subnets isolates activities such as test stations. Similar in form to an IP address. An example of a subnet mask value is 255.255.0.0.
Serial Target	This is the embedded device running kDiagnostics®, also called a device under test (DUT).

NOTE: Quoted text is paraphrased from Wikipedia*.

11
Contact Information

For additional support, contact Kozio:

Web site: www.kozio.com

E-mail:

- Technical Support: support@kozio.com

Phone:

- Sales: +1 (303) 776-1356 x1
- Technical Support: +1 (303) 776-1356 x2
- Fax: +1 (303) 776-1357

License Registration: www.kozio.com/license/